

FINAL REPORT FOR RESEARCH AGREEMENT

#INT-94979-RJVA

"Enhance MAGIS"

UNIVERSITY OF MONTANA

FS Contact: Dr. J. Greg Jones

CoOp Contact: Dr. Hans Zuuring & Ken Wall

Final Report

for

"Enhance MAGIS" Project (INT 94979-RJVA)

Prepared
by

Dr. Hans Zuuring
School of Forestry
The University of Montana
Missoula, MT 59812
(406-243-6456)

June 5, 1997

INTRODUCTION

For a number of years researchers at the School of Forestry, The University of Montana (UM), USDA Forest Service - Intermountain Research Station (INT), and Montana Department of Natural Resources and Conservation (DNRC)- Forestry Division have been developing computer software for planning land management and transportation-related activities on a geographic and temporal basis in the presence of multiple and sometimes conflicting objectives. This computer software was named **MAGIS** which is an acronym for **M**ulti-Resource Analysis and **G**eographic Information System. This system utilizes graphical and tabular data from an external GIS, together with numeric data entered via a series of user input screens. Once the necessary data have been entered the user can build planning scenarios through a SETUP procedure, obtain a solution to the problem and then display the results in tabular or graphical form. During the course of development of this software it became apparent that the data entry process and SETUP procedures could be greatly enhanced by constructing a graphical user interface (GUI) to streamline these tasks. To facilitate the construction of such a GUI a research joint venture agreement (INT 94979-RJVA) was drawn up between the USDA Forest Service - INT Research Station and The University of Montana in the amount of \$14,000 covering the period August 12, 1994 to September 30, 1996. This amount was matched with another \$10,000 from the Bolle Center for People and Forests, The University of Montana and \$1,920 in waived Indirect Costs for a total of \$25,920.

OBJECTIVES

The RJVA listed the following objectives:

- 1) Develop and implement Graphic User Interfaces for selected screens in the INPUT and SETUP components of MAGIS, including interface design specifications, developing the computer code to implement the interface, and testing and debugging.
- 2) Conduct BETA testing of Graphic User Interfaces and other components and debug as necessary.
- 3) Develop and implement an automated process for displaying user-selected attributes associated with treatment units (polygons) and road segments (lines) in both PAMAP and ArcView.
- 4) Complete user documentation, including a user guide and on-line help.
- 5) Disseminate research results.

METHODS

To meet Objective 1) a document containing a set of GUI specifications would be developed to guide the computer programmer who would generate the code. This document described a number of tasks, identified deliverables, and presented a schedule of fund disbursements. These specifications would be jointly developed by all three principal investigators namely: Will Wood (DNRC), Greg Jones (INT), and myself, Hans Zuuring (UM). The code would need to be tested and debugged with sample data sets. In addition, the executable code needed to be integrated seamlessly into MAGIS.

Sample data sets were given to the contractor to test and debug the GUI software so that Objective 2) could be achieved and the executable code was functioning properly and integrated into MAGIS.

Objective 3) would be met by writing macros for the PAMAP GIS software using the UC = xxxx KEYIN command and employing the Avenue scripting language in ArcView.

Objective 4) was part of an on-going effort to provide documentation and context sensitive help for users of MAGIS. This writing was to be conducted by Greg Jones and Hans Zuuring using WordPerfect or Microsoft Word and creating a populated .DBF file containing memo fields linked by keywords in FoxPro DBMS. The list of keywords would be jointly developed by all three principal investigators.

Objective 5) would be satisfied by making presentations at regional or national meetings and conferences.

RESULTS and DISCUSSION

From August 12, 1994 until January 12, 1995 the three principal investigators jointly developed the specifications for the construction of a GUI for MAGIS. These descriptions became part of a University of Montana contract that was subsequently awarded to HASTEC Computer Consulting, Inc. of Missoula, Montana on March 16, 1995 in the amount of \$18,648. The document containing the GUI specifications guided the private consultant so as to generate acceptable computer code to meet Objective 1) listed above. Appendix A contains a copy of the GUI specifications submitted to the private consultant. The actual work was performed by a subcontractor named Stuart Brittain from Manhattan, Montana who was an expert in writing graphics routines in C++ programming language.

From approximately April 1, 1995 until September 30, 1996 the GUI software was developed under the guidance of Scott Hassler from HASTEC and the above mentioned three project leaders. The GUI software was constructed through a series of meetings and periodic delivery of computer code which was subsequently tested to provide feedback to the contractor.

Sample data sets, both graphical and tabular, were made available to the contractor and the executable code was successfully integrated into the MAGIS menu system. The GUI code was able to read DLG files and open the associated database files. Various window operations could be performed such as zooming in and out and panning. Also labels as well as a legend box could be turned "on" or "off". During data input operations the user was able to aggregate treatment units into super treatment units and road segments could be combined to form model links interactively. Corridor sets could also be created.

In the SETUP phase of MAGIS the user was now able to click on a treatment unit and identify in a subsequent pop-up window which activities would be forced in solution. In the OUTPUT phase of MAGIS the user now has the option to interactively display which activities were initiated on any chosen treatment unit in a particular time period over the course of the planning horizon by clicking on the desired polygon label. All these GUI features make it easier for users to interact with MAGIS in a more intuitive manner.

The remaining \$5,352 was set aside to build digital line graph (DLG) files containing treatment unit polygons and road segments and to provide an automated method for displaying thematic maps containing planning model solutions from MAGIS in PAMAP GIS v4.2 and ArcView v2.1, a desktop mapping package from ESRI.

The results of this project have been disseminated to actual and potential users through demonstrations of MAGIS modified by the integration of the GUI at professional meetings, workshops and short courses.

APPENDIX A

**Montana Department of State Lands
Forestry Division**

**USDA Forest Service
Intermountain Forest and Range Experimental Station**

**University of Montana
School of Forestry**

**SPECIFICATIONS
FOR
MAGIS GRAPHICS INTERFACE**

**December 1993
(revised 1/12/95)**

TABLE OF CONTENTS

1.0 INTRODUCTION	1
2.0 BASIC GIS FUNCTIONS - OVERVIEW	1
2.1 Import Map Files in DLG Format	1
2.2 Map Element Manipulations	2
2.3 Query Graphic Element	2
2.4 GUI Main Menu Items	3
2.4.1 Background Graphics	3
2.4.2 Adding/Deleting Model Nodes	3
2.4.3 GUI Screen Design Considerations	3
2.5 Structure of GUI for SETUP	3
2.6.1 Assignment of Colors to Treatment Units	4
2.6.2 Assignment of Road Segment Line Styles/Colors	4
2.7 Reverse Previous Action	6
2.8 Legend Box Construction	6
3.0 RESOURCE PROJECT POLYGONS - OVERVIEW	6
3.1 Select/Exclude Super/Regular Treatment Units in SETUP	7
3.2 View Output	9
3.3 Creating a Super Treatment Unit	10
3.4 Deleting a Super Treatment Unit	11
3.5 Creation of Polygon Neighbor File	12
3.6 Management Regime Specs for Treatment Units	12
3.7 Variable Flow Loading in SETUP	14
4.0 ROAD NETWORKS - OVERVIEW	15
4.1 Data Capture of the Road Network (Data Links)	16
4.2 Initial Creation of Model Links	16
4.3 Addition of Model Nodes	16
4.4 Deletion of Model Nodes	17
4.5 Create New Model Link Level	18
4.6.1 Select/Exclude Road Projects in SETUP	18
4.6.2 Traffic Variables in SETUP	19
4.6.3 Final Demand in SETUP	20
4.7 Adding New Road Projects	21
5.0 OTHER DESIRED GRAPHICS ROUTINES	21
5.1.1 Create/Delete Wildlife Corridors	21
5.1.2 Select/Exclude Corridors in SETUP	22
5.2 Display Decision Variables in Solution	23
5.3 Create Display Mode item under MAGIS Main menu bar	25
5.4 Modifying Nongraphical Decision Variables Screen	25
6.0 APPENDIX A	26
6.1 APPENDIX B	27

1.0 INTRODUCTION

This document contains the specifications for a graphics front end to a custom modelling system to analyze land related activities (MAGIS). This system is being developed for Montana Department of State Lands, in cooperation with the USDA INT Forestry Sciences Laboratory and the University of Montana. The modelling system primarily deals with resource project polygons and transportation networks that service the planning unit.

The specifications outlined in this document deal with the design of a custom graphics interface which must execute independently of any GIS (Geographic Information System) software. Therefore some basic functionality inherent in a GIS must be built into the proposed interface. In addition, routines must be constructed to facilitate the loading of data (INPUT) for an area-specific planning model and the display of solutions (OUTPUT) spatially and temporally. These custom routines will permit the aggregating and color theming of treatment units, and road networks.

The specifications are described in three separate sections; one dealing with general GIS functions, another for loading and displaying treatment units, and the third for road networks.

2.0 BASIC GIS FUNCTIONS - OVERVIEW

Most GIS software packages contain a number of functions dealing with operations other than spatial analyses. These functions can be put into three groups: file conversion routines, map element manipulation query label routines. Each of these function groups is specified below.

2.1 Import Map Files in DLG Format

A routine needs to be written to import and export map files in DLG (Digital Line Graph) format so that vector elements can be displayed and manipulated. Three graphics files need to be imported namely **stand polygons**, **treatment unit polygons** and **network links**. This routine must be able to convert PAMAP DLG format files to DLG Option 1 format files. This latter format is a defacto industry standard for transporting map files from one GIS environment to another. This format consists of a header followed by a series of records containing information dealing with the map extent (min/max X,Y) and up to three different map feature records. These are **N** (node), **L** (line) and **A** (area) record types. Each specifies a tagID followed by a series of coordinates for nodes; node IDs for lines; and line IDs (+/- signs to denote left/right polygons) for areas. These sequences of records are repeated for all graphic elements on a map.

2.2 Map Element Manipulations

Once map elements (polygons, vectors and points) have been imported and displayed in a graphics window the user must be able to ZOOM IN and ZOOM OUT. Through the use of two magnifying glass icons, one containing a plus (+) and the other a minus (-), the user clicks on the desired icon and drags it to any location on the map. Upon releasing the **LEFT** mouse button the graphics scale increases or decreases by 50 percent. The user must be able to RESTORE the screen to the original scale as well as REFRESH the screen by clicking on an icon in each case.

The user must be able to ADD/DELETE/MODIFY labels which are linked to various map elements. In the case of a point the label and point are identical. In the case of a vector a new point is placed anywhere on the element which becomes a label and pointer to a record in a database if the structure of such a database has been previously defined. Similarly for polygons a new point is placed anywhere inside this closed area. This point represents a label and pointer to a record in a database whose structure has been previously defined. This functionality is needed for super treatment units and corridor routes. In all other cases it is sufficient to link database records to treatment units and model links through unique tagIDs. Appropriate databases (polygon, vector or point) must be opened to permit records to be added, deleted or modified.

The proposed interface must be able to color "fill" polygons based on a user defined color table (see Appendix A) and a selection/deletion function invoked by the user. This selection/deletion function would simply require the placement of the mouse cursor anywhere inside a polygon and clicking the **LEFT** button. In such a manner a series of polygons can be identified or "tagged" and upon completion the whole operation would be confirmed or "accepted" by clicking the **RIGHT** mouse button. The user must also be able to assign line styles to vectors (linear features like roads) which can be displayed at the same time as the above color "filled" polygons.

Another function should permit the querying of the labels or tags linked to map elements (points, vectors and polygons) so that the content of an associated database record can be displayed on the screen at the same time as the graphics.

2.3 Query Graphic Element

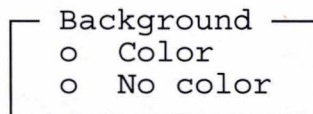
During the INPUT phase it is desirable to be able to show the initial conditions associated with any graphic element (polygon or vector) in a browse window whenever the user clicks on that element's **label**. Attributes stored in Tables 9a and 1 would be shown for network links while attributes stored in Table 18 would be displayed for treatment units. In addition the user should be able to display stand polygons and their labels (Table 16) overtop the treatment units and network

links that are in the background (subdued color) so that the user can click on these labels to display stand attributes.

2.4 GUI Main Menu Items

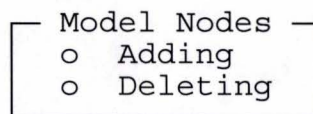
2.4.1 Background Graphics

From the menu bar of the GUI the user should be able to choose whether graphic elements in the background are to be colored or not. The user may want colored polygons and no colored network links or the reverse. The appropriate graphics layer will need it's colors turned off. In any case the following radio button panel needs to be created:



2.4.2 Adding/deleting Model Nodes

To permit the user to add and delete model nodes the following radio button panel needs to be created:



NOTE: If new records are added to/deleted from databases when in the graphics mode checks must be made to ensure that duplicate records do not occur.

2.4.3 GUI Screen Design Considerations

A prototype main graphics screen needs to be developed showing how radio buttons, control panels, menus, fonts, colors, etc. are to be incorporated into the GUI.

2.5 Structure of GUI for SETUP

From the MAGIS SETUP Menu the user clicks on the **Decision Variables** push button and resource project polygons appear in a graphics screen as a default. In the lower bottom right hand corner of this graphics screen a decision variables type panel containing five radio buttons appears as follows:

Type	
☐	Roads
☐	Traffic
☐	Res. Project
☐	Var Flow Load
☐	Final Demand

Done

NOTE: The resource project radio button is marked when this panel appears.

The user clicks on the desired radio button (if resource projects are not desired) and those graphic elements appear. When **Variable Flow Loading** is selected only those treatment unit polygons having loading options are displayed. Likewise, when **Final Demand** is selected only those model links attached to final demand nodes appear.

2.6.1 Assignment of Colors to Treatment Units

Treatment unit polygons are color filled on the basis of when land management activities occur over time. These color assignments are as follows:

Period of Implementation	Color	Color Code
1	green	4
2	blue	5
3	red	3
4	magenta	7
5	yellow	8

NOTE: The **no action** color, i.e. black, should be different from the background color, i.e. light grey.

2.6.2 Assignment of Road Segment Line Styles/Colors

The modelling routine needs to update the line styles for each of the road segments (model links). It is anticipated that the modelling program generates DBF files whose records contain codes for the road segment (model link) id and the line index value by time period. A line index value is a unique combination of line style and color.

Line Index	Line Style	Color	Period	Road Project
201	0 (solid)	2 (white)	all	existing
202	0 (solid)	4 (green)	1	reconstruct
203	0 (solid)	5 (blue)	2	reconstruct
204	0 (solid)	3 (red)	3	reconstruct

205	0 (solid)	7 (magenta)	4	reconstruct
206	0 (solid)	8 (yellow)	5	reconstruct
207	3 (dash/spc)	2 (white)	all	proposed NS ¹
208	3 (dash/spc)	4 (green)	1	proposed S ²
209	3 (dash/spc)	5 (blue)	2	proposed S
210	3 (dash/spc)	3 (red)	3	proposed S
211	3 (dash/spc)	7 (magenta)	4	proposed S
212	3 (dash/spc)	8 (yellow)	5	proposed S
213	1 (dotted)	4 (green)	1	obliterate
214	1 (dotted)	5 (blue)	2	obliterate
215	1 (dotted)	3 (red)	3	obliterate
216	1 (dotted)	7 (magenta)	4	obliterate
217	1 (dotted)	8 (yellow)	5	obliterate
218	8 (plus/spc)	4 (green)	1	closed
219	8 (plus/spc)	5 (blue)	2	closed
220	8 (plus/spc)	3 (red)	3	closed
221	8 (plus/spc)	7 (magenta)	4	closed
222	8 (plus/spc)	8 (yellow)	5	closed
223	4 (dash/dot)	4 (green)	1	temporary
224	4 (dash/dot)	5 (blue)	2	temporary
225	4 (dash/dot)	3 (red)	3	temporary
226	4 (dash/dot)	7 (magenta)	4	temporary
227	4 (dash/dot)	8 (yellow)	5	temporary

¹ construction not selected ² construction selected

Existing and **reconstructed** roads are to be coded as line style 0 (solid) with a specific color to denote the period of construction. Any **closed** road is to be coded as line style 8 (plus/spc) with a specific color to denote the period of closure. **Proposed** (constructed) roads are to be coded as line style 3 (dash/spc) with a specific color for period of implementation (see the table above as well as Appendix A). **Obliterated** existing roads are to be coded as line style 1 (dotted) with a specific color for period of implementation. **Temporary** roads are to be coded as line style 4 (dash/dot) with a specific color for period of implementation.

NOTE: It is desirable to have treatment units displayed in subdued colors as a backdrop when manipulating the road network. If this cannot be done then an additional five different line styles will have to be used in place of the above colors.

2.7 Reverse Previous Action

A function is needed to permit the user to cancel the selection/deselection of model nodes and member polygons of a super treatment unit. This could be accomplished by clicking on a graphic element label a second time with the (**LEFT**) mouse button. If not possible then a CANCEL menu item is needed which would cause a previous action to be reversed after clicking on the offending graphic element label. The idea here is to cancel a previous selection/deselection of any graphic element's label no matter what the original purpose was.

2.8 Legend Box Construction

Legend boxes need to be build to depict symbols, line styles/colors, line thicknesses, and color fills. The necessary data to construct such legends will exist in tables. Legends should be generated automatically and the location of such a legend box should be user-controlled. In addition, the user should have the ability to turn such a legend box "off" or "on" using the <**F10**> key as a toggle.

NOTE: Only show those features in a legend box which are active in the graphics screen.

3.0 RESOURCE PROJECT POLYGONS - OVERVIEW

The Resource Project Polygons are simply land management units, i.e. a collection of one or more whole or portions of timber stands. The objective is to determine during which periods (there are up to five planning periods) in the planning horizon each of the units are to be entered subject to a number of user-defined constraints. The activities imposed on these units are known as projects.

Sometimes it does not make sense to enter these units individually; it is desirable to combine some of these units into what is called a **Super Treatment Unit**. This combining of units is done so that additional roads may not have to be built, as well as other reasons. For example, if a unit along a road is harvested and the unit above it is not scheduled to be harvested until later, then a new road would have to be built to remove the logs from that polygon.

In the SETUP process the user can modify the planning model. Before the model is run, a user may wish to force certain units to be ignored or to force certain units to be entered in a specified period. This intervention is to be performed with the assistance of custom written graphics routines that allow the user to select polygons which are to be ignored or assigned to a specific project. When the user clicks on the **SET Decision Variables** task of SETUP followed by the selection of **Research Project** (Type 3 Variables) a graphics screen containing a current set of treatment units will appear (if **Graphical** mode selected under **OPTIONS** on MAGIS main menu bar).

For each polygon on the screen, there can be a variable number of records in the lookup table (Table 54). Each record is associated with a potential project for that polygon. When the user clicks on the tag of any polygon a scrollable window containing a list of all resource projects pertaining to that polygon over the whole planning horizon should appear on the graphics screen.

NOTE: We want to have other map features like roads or section lines displayed in the background (in a subdued color) while making management unit selections for super treatment units or corridor sets. The user should be able to make background features visible or invisible.

3.1 Select/Exclude Super/Regular Treatment Units in SETUP

During the SETUP process a custom function is needed which will allow a user to pick a polygon on a map which is to be **selected** or **excluded** when specifying a planning scenario. The polygon is selected by placing the cursor anywhere inside the polygon and pressing the data (**LEFT**) mouse button. Once the polygon is selected, a scrollable window appears which contains only those records associated with the polygon (from Table 54). The contents of this window is the same as the nongraphic mode and is as follows:

☐ Select ☐ Exclude ☐ Reset Done

-- Mgt Regime-- Exist Future	-- Log-- Meth	-- Impl-- Period	Propn Set	-- -- -- -- Status
SELECT	NULL	TRACTOR	1.00	Select ▲
SELECT	NULL	TRACTOR	0.00	Select .
CC	PCT	TRACTOR	0.00	Exclude .
CC	PCT	TRACTOR	0.00	Exclude .
CC	IP	TRACTOR	0.00	Exclude ▼

NOTE: This window and associated set of radio buttons should be the same as the current one in the non-graphical mode.

First the user clicks on one of the three buttons in the control panel located at the bottom of the window (ignore <Done> button for now) to indicate the desired status of each resource project selected by the user. A resource project is activated by clicking on its **Proportion Set** field. If the **Select** button was clicked then a value of 1.00 and the phrase '**Select**' appear in the Proportion Set and Status fields respectively of each activated resource project. If **ID_MSTR.Anal_Ty = 3** then only a value of 0.00 or 1.00 can be typed in the **Proportion Set** field. If otherwise, i.e. **ID_MSTR.Anal_Ty <> 3**

then any value in the range 0.00 to 1.00 can be typed in the **Proportion Set** field. If the **Exclude** button was clicked then as the **Proportion Set** field is clicked for each resource project that field is set to 0.00 and the phrase '**Exclude**' appears in the **Status** field. If the **Reset** button was clicked then as the **Proportion Set** field is clicked for each resource project that field is set to 0.00 and the **Status** field becomes blank. The user will only be allowed to modify the **Proportion Set** field. Once the user has modified all the desired resource project records, the program will perform the following edit checks:

1. For a given treatment unit displayed on the screen and Super Treatment Unit of which it is a member, the total of all resource project values in the **Proportion Set** field must not exceed 1. If this condition is violated display the following error message:

ERROR! Proportion set exceeds 1.

2. Under the **Status** field at least one entry must be blank or if there are NO blanks then the sum of all resource project values in the **Proportion Set** field must equal 1. If this condition is violated display the following error message:

ERROR! When all decision variables are set the proportion must sum to 1.

3. If any member polygon of a super treatment unit is set to a non-zero value the super treatment unit cannot be accessed. Display a warning message as follows:

WARNING! Cannot access super treatment unit because a member polygon has previously been set to a nonzero value.

4. If a super treatment unit has been set to a nonzero value its member polygons cannot be accessed. Display a warning message as follows:

WARNING! Cannot access member polygon because super treatment unit has previously been set to a nonzero value.

5. If have more than one project selected at a nonzero value for the same treatment unit use the color for the **earliest** period of implementation.

These checks are not performed until the user exits from the browse window and any errors must be corrected by the user before the data are permanently stored in the database records for this polygon (treatment unit).

Resource projects that have been selected should be displayed on the screen with the appropriate color fill to denote the period of implementation.

A procedure must be written to update a graphics database with color theme values that represent the periods that the projects are to be performed. The structure of this database will need to be modified to include two fields: the polygon ID and the color theme value. Both of these fields will need to be populated with values.

If **XXX_DV.Status** > 0 then obtain implementation period from Table 54 (COMP5) for that particular **Var_ID**. Then use appropriate color code (translation from implementation period to color code) to display that treatment unit's activity implementation period.

After clicking on [done] button at top of browse window shown in Section 3.1 then the color code for that treatment unit polygon must be updated based on the implementation period that was selected. If no resource project was selected turn polygon to background color.

Database Rules

1. The contents of XXX_DV.dbf is loaded into the Resource Project screen such that:
 - a) all resource projects (from Table 54) associated with the selected Treatment Unit are displayed in a scrollable window.
 - b) the **Proportion Set** and **Status** fields are updated with data from XXX_DV.dbf as follows:
 - i. If a decision variable is absent then the **Proportion Set** field is set to 0.00 and the **Status** field becomes blank.
 - ii. If XXX_DV.Status = 0 then place the phrase 'Exclude' in the **Status** field.
 - iii. If XXX_DV.Status > 0 then place the phrase 'Select' in the **Status** field.
2. The modified contents of the Resource Project screen needs to be written to XXX_DV.dbf as follows:
 - a) If **Status** field contains the phrase 'Select' or 'Exclude' then write the value in **Proportion Set** field to XXX_DV.dbf.
 - b) If **Status** field is blank then nothing is written to XXX_DV.dbf

3.2 View Output

This function permits the user to selectively display either treatment units or network links selected in a solution based on the color scheme rules mentioned in Sections 2.6.1 and 2.6.2. In addition upon clicking on the label of an individual treatment unit or network link a tabular report appears (overtop the graphics) in a browse window. This report shows the decision variables associated with the selected label that are in solution (see section 5.2).

This function is activated by clicking on **View** under the **OUTPUT** item

on the MAGIS main horizontal menu bar. At the start of this function the same menu screen appears as the nongraphical mode namely:

Scenario Soln	Previously Saved Scenario ID
- o Current - o Previous	
Graphic type	
- o treatment units - o network links	

The user clicks on either the current or previous scenario solution and selects an ID. Next the type of graphic elements to be displayed is chosen and the graphics appear. Then the user clicks on the label of a desired graphic element and a browse window appears showing the contents of OUT_COLS.dbf for a current solution or XXX_COLS.dbf for a previously saved solution.

3.3 Creating a Super Treatment Unit

This function will assist in the generation and definition of Super Treatment Units. The modelling program uses Tables 18 and 18a which list treatment unit polygons that, when combined, define a Super Treatment Unit. The purpose of this function is to allow the operator the ability to identify the resource project polygons that make up a Super Treatment Unit. This function will build the SUPRCUT table (Table 46a).

NOTE: if the **Polygon Neighbors** file is NOT present execute a routine to create it.

This function is executed by clicking on the [New] push button at which point the user is prompted to key in a unique Super Treatment Unit ID (not in Tables 18 or 46a). A check is needed to confirm that the entered ID is unique. Once entered, the user will be prompted to identify all the polygons that make up the Super Treatment Unit. This is accomplished by moving the cursor inside each polygon and pressing the data (**LEFT**) mouse button. The label of each treatment unit becomes bold once that polygon has been selected as a member of a Super Treatment Unit. Once all the polygons have been identified, the user will press the execute (**RIGHT**) mouse button. At this point the outer boundary of the super polygon should be denoted with a **bold** thick line. The definition process is finished when the [Done] push button is clicked in the control panel shown below:

New	Delete	Done
-----	--------	------

The program will perform edit checks on the super treatment unit definitions before the outer boundary is drawn.

NOTE: The companion non-selected graphic elements are displayed in subdued color.

There are two checks to be performed:

1. Make sure that all member polygons are neighbors. This is accomplished by first running the polygon properties function which determines the neighbors for all the polygons in the map. Later, in these specifications, a function will be described that builds the polygon neighbors file. We will assume that this function has been executed and that this table has been built. If the selected polygons comprising the Super Treatment Unit are not contiguous the following error message appears:

ERROR! One or more selected polygons are NOT contiguous in the current super treatment unit.

2. Make sure that each polygon is only part of one Super Treatment Unit. Polygons do not have to be part of any Super Treatment Unit. Check the SUPRCUT table to see if the polygon already exists. If any errors are found with the definition of a Super Treatment Unit, an error message is displayed on the screen as follows:

ERROR! XXX treatment unit is already a member of Super Treatment Unit ###"

where ### = _46A.super_treatment_unit.id

The user must determine the cause of the error and correct it. Once a Super Treatment Unit has passed the edit checks, the member polygon IDs and the Super Treatment Unit ID are written to the SUPRCUT table.

3.4 Deleting a Super Treatment Unit

This function will delete the definition of an existing Super Treatment Unit. There will be no functions to modify the definition of a Super Treatment Unit. Therefore, if the user wishes to change a Super Treatment Unit, they must first delete it, then add it with the updated definition.

The user clicks on the [Delete] push button and selects any polygon in the Super Treatment Unit. The program will determine the Super Treatment Unit ID and delete its definition from Table 46 (SUPRCUT). Once a Super Treatment Unit has been selected its boundary will flash. The user confirms the deletion by pressing the (RIGHT) mouse button. If the user selects a polygon that is not part of a Super Treatment Unit, the following error message appears:

ERROR! No Super Treatment unit defined at this location.

Another selection can be made by the user or click on the [Done] push button to exit this function.

NOTE: The user can cancel either the create or delete function at any time by pressing the <Esc> key.

3.5 Creation of Polygon Neighbor File

This function will normally be executed when the DLG files are imported into the GUI. A file is built which contains a list of neighbors (adjacent polygons) for each polygon on the map. This information will be stored in Table 18a named POLYNGBR. Each record in the POLYNGBR file contains two fields, a polygon id and one of its neighboring polygon's id. Therefore several records combined will inform the user of all the neighbors adjacent to a specific polygon. The user should be able to specify the amount of overlap required before polygons are said to be adjacent (Need to input and store this parameter).

If the user selected the Graphical mode from the **Display Mode** subitem under the **SYSTEM** item from the MAGIS main menu bar and subsequently manipulated Super Treatment Units or Adjacency Constraints then the system checks for the existence of this polygon neighbor file. If such a file does not exist then the user is prompted by the system as follows:

"Do you want to create a polygon neighbor file (Y/N/Q)?".

3.6 Management Regime Specs for Treatment Units

When the user clicks on the label of a super treatment unit or single treatment unit its boundary flashes, and a browse window appears containing the following items:

New	Delete	Done
-----	--------	------

Select

☐ Load Node

☐ Road Trigger

Mgt Regime		Logging Method	Time Period					Loading Type	Resource To Road Trig
Exist	Future		1	2	3	4	5		
SELECT	NULL	TRACTOR	T	F	F	F	F	Fixed	Required
SELECT	NULL	TRACTOR	F	T	F	F	F	Fixed	Required
CC	PCT	TRACTOR	T	F	F	F	F	Fixed	Suppress
CC	PCT	TRACTOR	F	T	F	F	F	Fixed	Suppress
CC	IP	TRACTOR	T	F	F	F	F	Variable	Required

Any of the above six fields can be altered by first clicking with the (**LEFT**) mouse button on the desired field to activate it. In the case of **Logging Method** and **Existing/Future Management Regimes** the user presses the <**F2**> function key to obtain a list of options in a pop-up box. Click on an option and the box disappears and the new value appears in the chosen field. The **True/False** designation for **Time Period** can be changed by first clicking on the field to activate it and then cycling through the available options. Clicking on the field a second time reverses the logical flag. For **Loading Type** there are two options: '**Fixed**' and '**Variable**' and the former is the default. For **Resource-to-Road Triggers** there are three options: '**Required**', '**Suppress**' and '**None**'. In each case the user clicks repeatedly on the field to cycle through the available options until the desired one appears. When the Loading Type is changed a warning message appears in the screen prompt box as follows:

"WARNING!! Changing the Loading Type will cause COST or PERCENT to be set to zero (Continue/Abort)?"

After the specification of the Loading Type the user must click on the **Load Node** radio button in the Select panel to make the pop-up window disappear and present a full view of the graphics screen. All existing model nodes (as identified from Table 46b) appear on the screen and blink. The user clicks on one model node and a small pop-up box appears with two columns labelled **Node ID** and **Cost** if the Loading Type was defined as '**Fixed**' or labelled **Node ID** and **Percent** if the Loading Type was defined as '**Variable**' (see Box Type A below). For Variable Loading the user must enter the **Proportion** of the selected resource project to be loaded onto that model node. Click on either the [**Done**] or [**Delete**] push button, located above the previously mentioned two-column box, to accept or reject the data value respectively. Repeat this process for all selected model nodes. Click on the [**Done**] push button to exit this function and perform the following check:

1. If the Loading Type is '**Fixed**' check to see that the sum of the percentages loaded over all selected model nodes equals 100 percent.

When the **Resource-To-Road Trigger** designation (Required or Suppress) is changed a warning message appears in the screen prompt box as follows:

"WARNING!! Changing the Resource-To-Road Trigger designation will require the identification of a **FROM** and **TO** node ID (Continue/Abort)?"

After the specification of the Resource-To-Road Trigger the user must click on the **Road Trigger** radio button in the Select panel to make the pop-up window disappear and present a full view of the graphics screen. If the Resource-To-Road designation is set to '**Required**' a small pop-up box appears with two columns labelled **FROM Node** and **TO**

Node (see Box Type B below). The user must select these two nodes from the graphics screen by clicking on their labels and then their IDs appear in the small box. Click on either the **[Done]** or **[Delete]** push button, located above the previously mentioned two-column box, to accept or reject the data value respectively.

Box Type A
[Delete] [Done]

NODE ID	COST or PERCENT

Box Type B
[Delete] [Done]

FROM Node	TO Node

NOTE: The last entry in the browse window can be copied and added to the list by clicking on the **[New]** push button.

3.7 Variable Flow Loading in SETUP

During the SETUP process when the user wants to set variable flow loading options and the graphical mode has been activated existing treatment unit polygons are displayed on the screen. The user clicks on the label of a polygon and a browse window appears showing the variable flow loading options for that polygon:

Done

Trt Unit	Load Node	Traffic Type	Time Period	Set to Zero?	
114	C14	T	1	F	▲
114	C14	R	2	F	.
					.
					.
					▼

Only the **Set to Zero?** field can be edited. Click on that field with the (**LEFT**) mouse button to activate it and again to reverse the logical flag from **False** to **True**. Initially all entries in this field are set to **False**. This process can be repeated by clicking on other

treatment unit polygon labels. Click on [Done] button to exit this function.

4.0 ROAD NETWORKS - OVERVIEW

This portion of the system deals with the existing and proposed road network associated with the treatment units. The road network will consist of many road segments (data links). The road segments will be grouped into larger pieces called model links. This is done to simplify and speed up the modelling process. Projects will be defined for these model links, much the same way projects were defined for the treatment unit polygons. The lookup table that holds the projects for the treatment unit polygons will also hold the projects for the model links.

The modelling process will classify the model links into several groups. These groups indicate when work is planned for the road segment. Each group will be themed for final output. The theming of the road networks will be accomplished by line styles (see Appendix A for definitions) rather than color. This is done so that polygons filled with colors can be displayed at the same time (in the background) as the road network. All these linear features will be displayed as white no matter what color they are assigned. Therefore the various types of road projects must be identified by line style.

The data capture and classification of the road network will be accomplished through several steps. The first step is to perform the data capture of the data links (smaller road segments). The next step is to generalize these data links into model links (larger segments). The second step will be an automated process that will define a large portion of the model links. The third step is to manually define additional model links, and to edit existing model links. The fourth step is to generate a network that only contains the model links. Once all the model links are defined, the fifth step is to generate records into Table 54 which represent projects for each model link. These five steps conclude the data capture portion.

Once the data is captured the user will wish to perform some analysis using the modelling system. There are several steps required to run the analysis. These steps are much like the definition of projects for treatment units. The user will manually define some projects for the model links. Projects for the model links will be things like initial road construction, reconstruction, road obliteration and closure, etc. The user will select or exclude projects for the model links, much the same way as selecting and excluding treatment unit projects. Once projects have been selected or excluded, the modelling program is executed. The modelling program needs the ability to modify the line styles for the model links. A special routine will need to be written to assist the modelling program in performing this task.

During the INPUT phase of MAGIS the user requires model nodes and their labels to create/delete model links. However, during the SETUP process model links and their labels are more important so model node labels should then be turned off.

4.1 Data Capture of the Road Network (Data Links)

The road network will be imported into the active graphics window at the same time as stand polygons via a DLG Option 1 Format translator. Node points will be input which delineate the appropriate road segments. A vector database tag is place on these road segments. Each node point is assigned a value which is four (4) alphanumeric characters in length. The road segment (data link) is uniquely named by the concatenation of the FROM and TO node identifiers.

4.2 Initial Creation of Model Links

There may a large number of data links on each map. This will slow down the modelling process, so it is desirable to combine the data links into larger groups called **model links**. The same logic and processes that are used in the nongraph-ical mode need to be used here. If the appropriate FOXPRO code can be called directly this is the preferred method otherwise this code will need to be converted into the C++ programming language.

NOTE: A unique code needs to be associated with each model node so that a special symbol (like a small solid filled square box) can be displayed on the screen for those nodes and regular point symbols (like a small triangle) for data nodes.

4.3 Addition of Model Nodes

A routine is needed to add new model nodes which create new **Model Links**. Users have the ability to decide where additional model nodes are to be placed. When this function is executed the user is prompted to identify a data node which they wish to become a new model node (i.e. the user will be identifying a node that will split an existing model link into two new model links). The identification of a node is performed by moving the cursor over top of the desired node and pressing the data (**LEFT**) mouse button. A special symbol will be placed at this location.. This lets the user know which nodes have already been selected. The function allows the user to select any number of nodes. Once the user has completed selection of the desired nodes, they press the (**RIGHT**) mouse button to exit the function.

Once the user has completed the identification of new model nodes, the program modifies the RDLOG file (Table 9) to incorporate the new model links defined by the nodes.

NOTE: It is probably desirable to have other map features like management unit polygons, section lines and contour lines displayed in the background (subdued color) during all model link operations.

4.4 Deletion of Model Nodes

A routine is needed to delete existing model nodes (ie. combine two model links into one). Users have the ability to decide which model links are to be deleted. When this function is executed the user is prompted to identify a model node which you wish to delete (i.e. you will be identifying a they node that separates two model links). The identification of a node is performed by moving the cursor over top of the desired model node and pressing the data (**LEFT**) mouse button. The selected model node blinks or pulsates to let the user know that it has been identified. The function allows the user to select any number of model nodes. Once the user has completed selection of the desired nodes, you press the (**RIGHT**) mouse button to exit the function.

As the user identifies each model node, the program performs the following edit checks:

- 1) Ensure the node selected by the user is indeed a **model** node.
- 2) Ensure the node selected by the user is not an intersection node. These nodes must remain as model nodes. This is done by checking the RDLOG table to determine the number of records that contain the node ID in either the FROM or TO model node fields. IF the count is 3 or more, then this is an intersection and is not allowed to be deleted.
- 3) Ensure that the data links on each side of the selected nodes have the same road option. If they do not, then the two model links on either side of the new model node cannot be merged into one model link. This is accomplished by checking the RDLOG table to ensure that any record which contains the new model node ID in either the FROM or TO node fields has the same road option value.
- 4) Ensure that the node is not a connection between an existing (**E**) and proposed/newly constructed road (**C**). If true then it cannot be deleted. Check to see that the LINK_COMP.Current.Status codes are the same!

If a node passes all the edit checks, the RDLOG file (Table 9a) is updated to reflect the new model links. When the function is complete, the selected model nodes stop blinking and the screen is redrawn. At this time the symbols for the selected model nodes become data nodes on the map. No flashing nodes should remain. If there are any present, this indicates an error has occurred while trying to delete a model link.

NOTE: The same logic and processes that are used in the nongraphical mode to perform the above edit checks need to be used here. If the appropriate FOXPRO code can be called directly this is the preferred method otherwise this code will need to be converted into the C++ programming language.

4.5 Create New Model Link Level

When all the model links have been defined and the operations associated with Sections 4.3 and 4.4 are finished a new level which contains only the model links that have been defined in the RDLOG table (Table 9a) will automatically be created. This new graphics level will contain only the model nodes and create the necessary vector tags for the model links.

This level is created to reduce graphic clutter as the user only needs to see the model links and their nodes, especially during the SETUP process.

NOTE: If Table 9a is altered and the graphic files exist a new model link graphic level needs to be created.

4.6.1 Select/Exclude Road Projects in SETUP

A function is needed to allow a user to pick a road segment on a map. The road segment is selected by placing the cursor over its label and pressing the data (**LEFT**) mouse button. Once the road segment (model link) is selected, a scrollable window appears which contains those records in Table 54 associated with this road segment. The contents of this window is the same as the nongraphic mode and is as follows:

Mode

☐ Select
 ☐ Exclude
 ☐ Reset

Done

- - -Road- - - -	Status	-Periods- Closed	-Impl- Period	Propn Set	- - - - - Status	
H	E	TTFFF	1	0.00	Exclude	▲
H	E	TTFFF	1	0.00	Exclude	.
H	E	TTFFF	1	0.00	Exclude	.
H	E	TTFFF	1	0.00	Exclude	.
H	E	TTFFF	1	0.00	Exclude	▼

First the user clicks on one of the three radio buttons in the **Mode** panel to indicate the desired status of each road project selected by

the user. A road project is activated by clicking on its **Proportion Set** field. If the **Select** button was clicked then a value of 1.00 and the phrase '**Select**' appear in the **Proportion Set** and **Status** fields respectively of each activated road project. If the **Exclude** button was clicked then as the **Proportion Set** field is clicked for each road project that field is set to 0.00 and the phrase '**Exclude**' appears in the **Status** field. If the **Reset** radio button was clicked then as the **Proportion Set** field is clicked for each road project that field is set to 0.00 and the **Status** field becomes blank. The user will only be allowed to modify the **Proportion Set** field. Once the user has modified all the desired road project records, the program will perform the following edit checks:

1. For a given model link displayed on the screen, the total of all road project values in the **Proportion Set** field must not exceed 1. If this condition is violated display the following error message:

ERROR! Proportion set exceeds 1.

NOTE: this check is not performed until the user exits from the browse window.

2. Under the **Status** field at least one entry must be blank or if there are NO blanks then the sum of all road project values in the **Proportion Set** field must equal 1. If this condition is violated display the following error message:

ERROR! When all decision variables are set the proportion must sum to 1.

These checks are not performed until the user exits from the browse window and any errors must be corrected by the user before the data are permanently stored in the database records for this model link.

Road projects that have been selected should be displayed on the screen with the appropriate line styles and colors which depend on the period of implementation.

4.6.2 Traffic Variables in SETUP

During the SETUP process when the user wants to set traffic decision variables and the graphical mode has been activated existing model links are displayed on the screen. The user clicks on the label of a model link and a browse window appears as follows:

Done

Dirn	-Traffic- Type	- - -Road- - - Option	Period	- -Closed- - To Traffic	
F	T	H	1	F	▲
B	T	H	1	F	.
F	R	H	1	F	.
B	R	H	1	F	.
F	T	H	2	F	▼

Only the **Closed To Traffic** field can be edited. Click on that field with the (**LEFT**) mouse button to activate it and again to reverse the logical flag from **False** to **True**. Initially all entries in this field are set to **False**. This process can be repeated by clicking on other model link labels. Click on [**Done**] button to exit this function.

NOTE: The last entry in the browse window can be copied and added to the list by pressing the **DOWN ARROW** key.

4.6.3 Final Demand in SETUP

During the SETUP process when the user wants to set final demand nodes and the graphical mode has been activated, existing model links containing final demand nodes and their labels are displayed on the screen. The user clicks on a label and a browse window appears as follows:

Done

Final Node	Traffic Type	Time Period	Lower Limit	Upper Limit	Reln Set?	
049	T	1	0.0	0.0	F	▲
049	T	2	0.0	0.0	F	.
156	T	1	0.0	0.0	F	.
156	T	2	0.0	0.0	F	.
078	T	1	0.0	0.0	F	▼

Only the **Reln Set?** field can be edited. Click on that field with the (**LEFT**) mouse button to activate it and again to reverse the logical flag from **False** to **True**. Initially all entries in this field are set

to **False**. This process can be repeated by clicking on other model link labels. Click on [**Done**] button to exit this function.

4.7 Adding New Road Projects

It is desirable to have a graphical version of screen set 9a for the purpose of adding new road projects to selected model links. The user clicks on the label of a model link and a browse window appears containing road options for that link as follows:

New	Delete	Done
-----	--------	------

From	To	Type	Option	Speed	Grade	Ad_length
001	049	4	H	0.0	0	1.36
Select New Rd Option				.0	-5	0.27
Alignmt	Surface	Lanes	Option	.0	-5	0.59
				.0	-5	0.58
GOOD	GRAVEL	1	H	.0	-5	0.35
EXCEL	PAVED	1	H	.0	-5	1.22
GOOD	PAVED	1	H	.0	-2	0.15
FAIR	PAVED	1	H	.0	-5	0.60
GOOD	TEMP	1	T	.0	2	0.14
				.0	0	1.04
				.0	-10	0.14

Click on the [**New**] push button and a second pop-up window appears overtop of the first browse window containing valid road options not previously selected for that link. The user selects a road option. To process another model link the user clicks on the [**Done**] push button. To eliminate a road option highlight the offending line in the first window and click on the [**Delete**] push button. This approach is the same as that of screen 9a in the nongraphical mode with the exception that only one model link is operated on at a time.

NOTE: The functionality in the browse window is the same as the non-graphical mode.

5.0 OTHER DESIRED GRAPHICS ROUTINES

5.1.1 Create/Delete Wildlife Corridors

We need to define corridors of connecting treatment units between any

two locations (polygons) by clicking on a series of polygons between these locations. The name of the overall region, the names of each possible corridor of connecting polygons as well as the polygon IDs making up each corridor need to be stored in a database file. The chosen polygons forming a wildlife corridor must be connected and have vegetation in acceptable successional stages as defined by the user. To generate corridor sets the same procedure is used as that for Super Treatment Units, i.e. bold label when individual polygons are selected and use a bold thick line to denote the outer boundary of the polygons comprising the completed corridor. Unlike Super Treatment Units an individual polygon can be a member of more than one corridor set.

New	Delete	Done
-----	--------	------

After clicking on the [New] push button the user chooses the length of time for the existence of the route by selecting either **P** for time period or **H** for the whole planning horizon. Next the user chooses the acceptable successional stages by clicking on a valid name list. Then an active corridor set name is defined (typed) and placed in Table 12a or else selected from a list of candidate names. Next the route name is defined (typed) and placed in Table 12b with the internal code from Table 12a for that corridor set. A route is built by successively clicking on a series of treatment unit labels comprising the corridor route using the (**LEFT**) mouse button and clicking the (**RIGHT**) mouse button when finished. Next the user must click on any map location wholly inside this corridor route to place the route ID label. If the [Delete] push button is clicked corridor routes are deleted by clicking on one or more corridor route labels with the (**LEFT**) mouse button and executing this function by clicking the (**RIGHT**) mouse button.

NOTE: Tables 12a and b must be checked for the non-existence of the new corridor set and route names to prevent duplicates!

5.1.2 Select/Exclude Corridors in SETUP

During the SETUP process the user needs to be able to make decisions about the selection of existing corridors by time period. This is a multi-step process: click on **Special Relns** push button in SETUP menu and on the next screen make corridors active by clicking its radio button. Then click on **Corridor Set Selection** push button in the Access Window for Corridors and the existing corridor routes and their labels appear on the graphics screen. Next the user clicks on one of the labels (it flashes) and the Action and Time Period panels appear:

Action ☐ Select ☐ Unspecified ☐ Exclude	Time Period ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5	Done
--	--	---

Next the user clicks on a Time Period (this is active only when length of time for corridors = 'P') radio button followed by a selection from the Action panel. Clicking on the Select radio button causes the corridor to be included in future analyses and the corridor is filled with a green color. If the Unspecified radio button is clicked the corridor is not selected by the user and is filled with the background color. If the Exclude radio button is clicked the corridor is removed from consideration and is filled with a red color. Click on the **[Done]** push button to terminate the Select/Exclude Corridor function.

5.2 Display Decision Variables in Solution

During the OUTPUT phase the user needs to be able to click on labels of treatment units as well as those of model links and display their non-zero activities or show the contribution of individual decision variables to specific management relations from either XXX_COLS.dbf or OUT_COLS.dbf in a browse window. The following panels appear:

Display Attribute ☐ Proportion in Solution ☐ Contribution to Mgt Relns		
Type ☐ Road Project ☐ Traffic ☐ Res. Project ☐ Var Flow Load ☐ Security Areas ☐ Corridors	Time Period ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5	Done

First the user must select which solution attribute is to be displayed by clicking on either the **Proportion in Solution** or **Contribution to Mgt Relns** radio button in the Display Attribute panel. Next the type of decision variable is selected by clicking on the appropriate radio button in the Type panel and only those colored graphic elements and their labels associated with this choice appear. Colors are assigned

according to the rules in Sections 2.6.1 and 2.6.2.

If the **Proportion in Solution** radio button is clicked then for any of the first three types of decision variables (road project, traffic and resource project) the graphic elements and their labels over all time periods are displayed. The last two types of decision variables (security areas and corridors) display graphic elements with no labels after a radio button in the **Time Period** panel is clicked. Either the boundaries of security areas and corridors should be shown or the spatial arrangement of such entities could be shown by color filling all the member polygons. The same algorithm as utilized by GIS Output (selected from MAGIS Main menu bar) should be used.

The selection of variable flow loading causes only those treatment units (and their labels) having variable flow options to be shown.

Upon clicking on a graphic element label a browse window (in this case for resource projects) appears:

Done

Mgt Exist	Regime Future	Logging Method	Time Period	Solution Value	
SELECT	NULL	TRACTOR	1	0.65	▲
SELECT	NULL	TRACTOR	2	0.15	.
CC	PCT	TRACTOR	1	0.00	.
CC	PCT	TRACTOR	2	0.20	.
CC	IP	TRACTOR	1	0.00	▼

NOTE: The format of this browse window should be consistent with the current output tables.

The contents and headings of the browse window change depending on the type of decision variable selected. In any event the contents can only be viewed and not edited. Click on [**Done**] push button to select another graphic element label.

If the **Contribution to Mgt Relns** radio button was clicked only the first three decision variable types are available from the Type panel. After the selection of a decision variable type only those colored graphic elements and their labels associated with this choice appear. Upon clicking on a graphic element label a browse window (in this case for resource projects) containing all management relations pertaining to the selected graphic element are listed as follows:

Done

Mgt Reln Type	Name	Contribution	
			▲ . . ▼

Click on [**Done**] push button when finished looking at browse window and to select another graphic element label.

NOTE: Entries in the above browse window are sorted by relation type and values in the Contribution column are summed over all decision variables in solution for that graphic element. A contribution value of an individual decision variable = its activity level x its coefficient.

5.3 Modifications to **MAGIS** Main menu bar

- a) A **Display Mode** submenu item needs to be added under the **SYSTEM** item on the **MAGIS** Main Menu bar which permits the user to select the graphical or non-graphical (tabular) mode for certain applicable data INPUT, SETUP and OUTPUT processes. When the graphics option is selected, the software checks for the presence of the graphic files (already loaded into the GUI). If they are not complete the user is asked:

"Do you want to import the DLG files (Y/N)?".

If the response is **N**, the graphic option cannot be turned-on. If the response is **Y** the DLG files are processed. If the importation is successful (ie. no errors) then the graphics mode is activated. If not successful, the error messages are written to an error file and the user is informed that: 1) importation was not successful, 2) the location of the error log file and 3) that the graphic mode cannot be activated until the import is successful.

- b) Under **OUTPUT** item on the **MAGIS** Main Menu bar a third submenu item called **View** needs to be added so that solution information can be interactively displayed for user-selected graphic elements. This item should only be enabled when the Display Mode has been set to graphical.

5.4 Modifying Nongraphical Decision Variables Screen in SETUP

When a user selects the **Set Decision Variables** task in SETUP it takes a long time for the program to build a temporary file containing activities associated with all resource and road projects. In the nongraphical mode a pop-up box containing valid project IDs is needed after the user clicks on the resource or road project button in order to be consistent with the GUI. From this browse window the user selects a project ID and then another pop-up window appears containing the associated activities. In this manner only the pertinent activities associated with a single selected project appear.

6.0 Appendix A

Color Definitions

To display color background fill for treatment units and assign colors to line styles standard color codes must be defined. Each color is assigned a numeric code from 1 to 16 as defined below:

1 => BLACK	9 => ORANGE
2 => WHITE	10 => GREEN YELLOW
3 => RED	11 => GREEN CYAN
4 => GREEN	12 => BLUE CYAN
5 => BLUE	13 => BLUE MAGENTA
6 => CYAN	14 => RED MAGENTA
7 => MAGENTA	15 => DARK GREY
8 => YELLOW	16 => LIGHT GREY

Colors are defined as either the RGB (Red, Green, Blue) or HLS (Hue, Lightness, Saturation) type. Colors can be altered by changing the Hue (0 - 360) or Red (0 - 100) value, the Lightness or Green value (0 - 100), and the Saturation or Blue value (0 - 100). The above 16 color assignments are stored in a color definition table.

Line Styles

Each line style is assigned a numeric code from 0 to 7 as defined below:

0 - solid	4 - long dash, dot
1 - dotted	5 - short dash/space
2 - short dash/long space	6 - short dash, 2 dots
3 - long dash/short space	7 - long/short dash
	8 - plus/short space

6.1 Appendix B

Table 54: VARNAMES

VARIABLE NAMES							
Inter. <u>VAR_ID</u>	Variable <u>Type+</u>	----- Variable Name* -----					
		<u>Comp1</u>	<u>Comp2</u>	<u>Comp3</u>	<u>Comp4</u>	<u>Comp5</u>	<u>Comp6</u>
5N	1C	4C	4C	3C	3C	3C	1C

+ 1 = roads, 2 = flow, 3 = res. project, 4 = variable flow loading, 5 = final demand
6 = security area variable, 7 = corridor variable, 8 = unroaded variable

* up to six components (each of varying width) dependent on variable type as follows:

<u>Var Type</u>	# of nonblank <u>chars</u>	<u>Description</u>
1	13	variable type (1C) + node 1 (4C) + node 2 (4C) + blank + road status (1C) + road standard (3C) + period of imple (1C) WHERE road status is defined as follows: C = new construction, R = reconstruction, N = no activity, E = Existing
2	13	variable type (1C) + node 1 (4C) + node 2 (4C) + direction (1C) indicator for traffic type (3C) + road standard (3C) + period (1C)
3	10	variable type (1C) + cutting unit ID (3C) + existing MR (3C) + future MR (3C) + logging method (3C) + period of imple (1C)
4	13	variable type (1C) + ID for the dummy node coded as \$XXX (XXX = the polygon ID for the resource project) (4C) + ID for node b (4C) + direction code 'F' (from \$XXX to node b) + traffic code (3C) + blank + period of traffic (1C)
5	7	variable type (1C) + Node b (final demand node) (4C) + blank (5C) + traffic code (3C) + blank (1C) + period of traffic (1C)
6	6	variable type (1C) + internal code in Table 12b corresponding to the Discrete class name for Security Area k in Table 55 (4C) + set number (period number [if security relations are by period] or letter 'H' [if security relations are for planning horizon]) (1C)
7	6	variable type (1C) + internal code in Table 12b corresponding to the Discrete class name for Corridor Route k in Table 55 (4C) + internal code in Table 12a corresponding to Corridor Set Name (3C) + set number (period number [if corridor relations are by period] or letter 'H' [if corridor relations are for planning horizon]) (1C)
8		variable type (1C) + cutting unit ID (4C) + period # (4C)

Final Report

for

“Vegetative Pathways in MAGIS” Project

Phase I

(INT 96038-RJVA)

Prepared
by

Dr. Hans Zuuring
School of Forestry
The University of Montana
Missoula, MT 59812
(406-243-6456)

June 10, 1997

INTRODUCTION

For a number of years researchers at the School of Forestry, The University of Montana (UM), USDA Forest Service - Intermountain Research Station (INT), and Montana Department of Natural Resources and Conservation (DNRC)- Forestry Division have been developing computer software for planning land management and transportation-related activities on a geographic and temporal basis in the presence of multiple and sometimes conflicting objectives. This computer software was named **MAGIS** which is an acronym for **M**ulti-Resource Analysis and **G**eographic Information **S**ystem. This system utilizes graphical and tabular data from an external GIS, together with numeric data entered via a series of user input screens. Once the necessary data have been entered the user can build planning scenarios through a SETUP procedure, obtain a solution to the problem and then display the results in tabular or graphical form.

Over the last year a number of MAGIS users have asked for an alternative approach to projecting vegetation over time that is more compatible with ecosystem management at landscape scales. This new approach involves the use of vegetative states which move along certain pathways over time in the presence of natural processes. The standard method of vegetation projection utilizes timber strata where trees are characterized by DBH, height, basal area, and volume. Using an optional tree growth simulator like SPS or PROGNOSIS (FVS) the above attributes are modified over time in the presence or absence of management activities. This latter method of vegetation projection is the only one currently incorporated into MAGIS and does not lend itself to ecosystem management. To add this additional capability to MAGIS a research joint venture agreement (INT 96038-RJVA) was drawn up between the USDA Forest Service - INT Research Station and The University of Montana in the amount of \$24,000 covering the period March 29, 1996 to March 31, 1997. This amount was matched with another \$4,800 in waived Indirect Costs from the University of Montana for a total of \$28,800.

OBJECTIVES

- A. This agreement continues MAGIS-related research and development efforts initiated under Cooperative Research and Development Agreement INT 92655-CRADA, "MAGIS Agreement" which expired on June 30, 1993, and Research Joint Venture Agreement INT 94979-RVJA, "Enhance MAGIS", which ended September 30, 1996.
- B. This open-ended agreement provides for formal interaction on the following Phase I specific objectives within the overall, joint research and development project:
 - 1) Develop and implement a vegetative projection approach that uses vegetative states and pathways describing change among these states, i.e. the duration of years required to transform various classes of vegetation from one state to another given the presence or absence of management activities or natural processes. This includes outlining the database structures and logic, designing the user interfaces,

developing the computer code to implement the approach, and testing and debugging.

- 2) Conduct BETA testing of the vegetative projection approach and other related components and debug as necessary.
- 3) Complete user documentation, including a user guide and on-line help.
- 4) Disseminate research results.
- 5) Test the use of a LANDSAT-derived vegetation data layer and associated attributes in MAGIS and compare planning model results with the use of TSMRS/PI data, as planned under Objective 1) of the Bitterroot Ecosystem Management Research Project (BEMRP) Landscape Analysis Group.
- 6) Build and run MAGIS models for testing formulation and data aggregation techniques while conducting analyses for Planning Units ranging in size from 200,000 to 500,000 acres, as stated under Objective 2) of the BEMRP Landscape Analysis Group.

METHODS

To meet Objective 1) under item B. listed above, a document containing a set of specifications for developing a vegetative pathway projection approach was created to guide the computer programmer who would generate the necessary code. This document described a number of tasks namely: develop a new process to import pathway data; create new data input screens; modify existing data input screens and associated data tables; and program new batch processes. In addition, deliverables were identified, and a schedule of fund disbursements was given. These specifications were jointly developed by two researchers namely: Will Wood (DNRC), and Greg Jones (INT).

New model formulations were created and solved by Will Wood, Greg Jones and a new computer programmer to test and debug the vegetative pathway projection approach so that Objective 2) could be met and the executable code was functioning properly and integrated seamlessly into MAGIS.

Objective 3) was part of an on-going effort to provide a MAGIS user guide. This writing was to be conducted by a temporary employee who would be supervised by Greg Jones and Hans Zuuring.

Objective 4) would be satisfied by making presentations at regional or national meetings and conferences.

Objectives 5) and 6) would be accomplished by Will Wood (DNRC) and Greg Jones (INT) by formulating and solving planning scenarios for larger geographic areas.

RESULTS and DISCUSSION

From March 29, 1996 until August 30, 1996 the two researchers mentioned above jointly developed the specifications for incorporating a vegetative pathway projection approach (VPPA) into MAGIS. These descriptions became part of a sole source USDA Forest Service contract that was subsequently awarded to HASTEC Computer Consulting, Inc. of Missoula, Montana on September 30, 1996 in the amount of \$18,090. The document containing the VPPA specifications served as a guide to the private consultant so as to generate accept-able computer code to meet Objective 1) listed above. Appendix A contains a copy of the VPPA specifications submitted to the private contractor.

From approximately October 1, 1996 until March 31, 1997 numerous modifications were made to MAGIS that involved changing existing data input screens and the creation of new ones and associated databases. In addition, a procedure for importing pathway data into MAGIS and the programming of batch processes was completed. Scott Hassler from HASTEC coded most of the changes in MAGIS using the FoxPro programming language. Will Wood and a newly hired computer programmer by the name of Kurt Krueger also wrote code and assisted in the testing and debugging phase.

In the BUILD MODEL phase of MAGIS the new and modified data input screens were examined and exercised to see if they functioned properly. The user now has two choices for projecting vegetation over time: the standard timber strata approach *and* the pathway approach.

Considerable progress was made on a user guide as well as training materials by hiring a temporary writer/biologist named Janet Sullivan starting last October 1996. The on-line help in MAGIS was completed and a tutorial data set was built. In addition, two training sessions were held for personnel from both the Colville and Salish and Kootenai Tribal Agencies. One of these sessions was a 2-day introduction to MAGIS for managers held last January in Missoula while the other was a 4-day short course about the mechanics of MAGIS and how to formulate planning scenarios with lots of hands-on exercises for field foresters. Both were a big success!

The results of this project have been disseminated to actual and potential users through demonstrations of MAGIS modified by the integration of the VPPA at professional meetings, workshops and short courses. We also have "cut" a CD-ROM containing a setup procedure for installing MAGIS under Windows 95 with all the necessary executable code, sample data sets, and a licensing agreement.

Objective 5) was postponed until Phase II of this project since the classified LANDSAT imagery was not available. Therefore, vegetation data obtained from such an image could not be tested against TSMRS/PI data.

Likewise, Objective 6) had to be postponed until Phase II of this project because we were unable to acquire all the necessary data.

APPENDIX A

Technical Specifications
for
Vegetative Pathway Projection
Contract

Prepared
by
Greg Jones and Will Wood

August 30, 1996

1. Import routines and associated screens

Item 1.1 Program a routine to import vegetative pathway data and populate the following data tables.

State Definition Table

def_state.dbf (INTSTATE N(4,0), DOM_SP_I n(4,0), SZ_CLASS_I n(4,0), DENSITY_I n(4,0)).

INTSTATE - Internal code for state

DOM_SP_I - Internal code for dominate species (_12b)

SZ_CLASS_I - Internal code for size class (_12b)

DENSITY_I - Internal code for density (_12b)

State Process/Activity Table

pro_state.dbf (INTSTATE n(4,0), HBTY_GP_I n(4,0), time_inc n(4,0), INT_ACT_ID n(4,0), RESTATE n(4,0), Rtime_inc n(4,0)) (we still need to have all the variables that define a state be available for output and cost calculates).

HBTY_GP_I - Internal code habitat type group (_12b)

TIME_INC - time increment in state

INT_ACT_ID - Internal code for activity (_37)

RESTATE - Internal code for the resulting State

Rtime_inc - Time increment in the resulting State

Specifications for the file to import into pro_state.dbf:

File name - user specified (hbty_grp c(20), Dom_sp c(20), sz_class c(20), density c(20), time_inc c(10), activity c(20), r_dom_sp c(20), r_sz_class c(20), r_density c(20), r_time_inc n(4,0))

hbty_grp - Habitat type group name

dom_sp - Dominate Species name

sz_class - Size class name

density - density class name

time_inc - time increment in state

activity - activity/process name

r_dom_sp - resulting dominate species

r_sz_class - resulting dominate species

r_density - resulting density

r_time_inc - resulting time increment

Specifications for populating the State Definition Table and State Activity Table:

Allow the user to select the name to use for the file to import to populate tables def_state and pro_state. Ask the user if they want to replace the current state definition table or the state process table or append to these tables. First, select all the distinct combinations of dom_sp, sz_class, & density, and r_dom_sp, r_sz_class, & r_density. Check for duplicates in the importation process (either internal to the imported file (across both combinations) or between the imported file and current states in def_state) and give an error message if duplicates are found. Convert the imported name to MAGIS names (if a cross walk

table is provided), then convert the MAGIS names to internal codes, define int_code for state, and append the records to def_state. If names are not found in _12as or _12b or in the crosswalk table (if present), give an error message and abort the importation process (check the entire file for errors, if processing is too slow, stop after 200 errors). Time increment length in the pathway data must equal the length of the time units as specified in table _32.

Item 1.2 Crosswalk Table. If the names used to define state are different from the MAGIS _12b names, allow the user to use a crosswalk table. The user selects a file that will be a crosswalk between: the names of habitat type group, dominate species, size class, and stocking, and the 12b names in MAGIS. The cross walk table should also include activity ids, with relate to table _37). The structure of the imported file will be three fields delimited by commas:

Field 1: the imported name maximum character width 20,
Field 2: the MAGIS _12b name maximum character width 8,
Field 3: the name type code. The valid type names are HBTY_GRP, DOM_SP, SZ_CLASS, DENSITY and ACTIVITY.

Display error messages for other names that don't match. First check if there is a corresponding _12as.name that matches the type code name from the crosswalk table (for all types not equal to ACTIVITY). If not add the type code to _12as & _12a (HBTY_GRP, DOM_SP, SZ_CLASS, DENSITY). Create a new internal code to add to _12b (no duplicates). If the type is ACTIVITY and not found in table _37 activity name add it to _37 with a activity type equal to ' '.

2. Create New Screens

Item 2.1 Program screen to define/edit States (i.e. build table def_state). The basic layout for the screen follows:

Delete	Done
--------	------

Habitat Type Group

 :

Dominate Species

 :

Size Class

 :

State : (update state base on which density row
the user is at.)

	Time Increments (decades) within state									
Density	1	2	3	4	5	6	7	8	9	10
xx1	X	" "								
xx2	" "									
xx3										

State is defined as Dominate species + '/' + size class + '/' + density. where xx1,xx2, and xx3 are _12b.name for density. Habitat Type Group, Dominate Species, and Size class are push buttons that result in all the valid _12b names for the given _12a name. Display all combinations of density (all the _12b names) and 10 time increments. Write to def_state all new states based on the dominate species, size class, and all density classes. The user must define all earlier time increments within a density class to access a time increment in the row.

No duplicates. Delete removes the state from def_state and pro_state (all records associated with the deleted state). Give warning with option to delete for current density row (default to row 1) or all densities row. Default the choice to the current row. The cells in the bottom portion of the screen identify the combinations of Density and Time Increment for which activity information has been entered (this is done on a screen which is accessed from this screen). X means that activities information has been entered, and " " (blank) means no information regarding activities on has been entered. The screen for entering activity information is accessed by pressing <F12> or "enter" and the following box appears:

Action on: "State"

Edit/Create

Cancel

Copy

"State" represents the actual State Name. The Edit/Create option accesses the Activity/Process Screen, Item 2.2. The copy option is used to copy information previously entered for a combination of density and time increment to the current combination. Copy brings up a box for selecting what to copy from: State, Density, and Time Increment, and includes an option to cancel out of copy mode.

Item 2.2 Program the State Activity/Process Screen, which has the following form.:

Done
Cancel

Current State :

Dominate Species:

Stocking:

Time Increment:

Filter ()
Locate ()

Dom Species

Size Class

Density

DF
Large

Process/Activity	Next State for Process/Activity
SUCCESSION	(either type in state & time increment or <f12> pop-up a
Activity 1	list of states & time increments
Activity 2	using the filter to subsample
L-MPB	list or locate to go to selected
L-S-FIR	attributes. Order list by dom
	species, size class, density, time
	increment)

Process/Activity

Add
Delete

Process/Activity: <Add> produces a pop-up list all the activities from _37.

<Delete> removes a state & time increment, activity combination and resulting state & time increment from pro_state. Give a warning before deleting with option to delete the highlighted process/activity (default to the first process/activity) or all the process/activity defined for this state.

If the Done button is selected, and the process "SUCCESSION" has been defined, add to pro_state table for each combination of valid current state & time increment, process/activity and state & time increment. No duplicates. If the user did not define "SUCCESSION" give warning message "Please Define Future State based on SUCCESSION. OK Cancel" (OK returns to screen, Cancel abort to Edit/Create State screen).

Item 2.3 Program a screen for Define/Edit State Group Name

New Tables

_43p (in_mgrg_id n(3,0), tu_min n(2,0), tu_max n(2,0), int_grp n(4,0))
in_mgrg_id - internal management regimes ids from _43b.
tu_min - Minimum time units into specified state group
tu_max - maximum time units into specified state group
int_grp - internal group code

_43pa (int_grp n(3,0), name_grp c(8))
int_grp - internal group code
name_grp - group name

_43pb (int_grp n(3,0), HBTY_GRP_i n(4,0), intstate n(4,0))
int_grp - internal group code
hbty_grp_i - internal habitat type group code
intstate - internal state code

Screen Title - Define/Edit State Group Name

New	Delete	Edit Name	Done
-----	--------	-----------	------

List of
State Groups

--

Candidate
Hab Ty Grp + State

--

Selected
Hab Ty Grp + State

--

Ok	Abort
----	-------

New - defines a group name, int_grp number and adds a new record to _43pa. Check for duplicates.
Delete - deletes all records from _43pa, _43pb and _43p that have same int_grp number.
Edit Name - allows the user to edit the group name stored in _43pa.name_grp.
The list of candidate Hab Ty Grp + State is from def_state table.
Ok push button saves int_grp, hbty_grp_i, and intstate to _43pb for all selected Hab Ty Grp + State.
Abort does not update _43pb at all. Give warning message when aborting.

Item 2.4 Program a screen for Management Regime Pathway / Update table 43b.

Screen Title - Management Regime Pathway

New MR	Delete MR	Done
--------	-----------	------

List of
MR Names

--

TU's into
Specified State

Process/Activity	Min	Max	<State Group Name>
Row A (1)	(2)	(3)	(4)

Activity Following	
New	Delete

Following Activity TU's from
First Activity

Row B (5)

(6)

New MR are stored to _43b.name and create a new internal code. Delete MR removes them from _43b and call the appropriate del_tree routine. Only one record in row A, allow user to select activity from a pick list (do not allow user to type in this field). Store to table _39, the in_mgrg_id for the current MR, internal activity code from (1), and 0 in tu_impl. There must be a valid activity name in (1) before the user can add activities to row B. New Act and New Following push buttons yield the list of activities from _37. For the New Following push button, add activity to row B (more than one row allowed). Delete Following deletes the current record from row B. Delete Act deletes activity in (1). <State Group Name> yields a list from table _43pa name_grp. If none, give a warning message. All Following activity (5) and TU's from first activity (6) and the current internal code for MR are stored to _39.

Item 2.5 Program a screen for second level specification for assigning MR to treatment units

- A) Specify Zone Attribute name (_12a)
- B) Specify associated zone attribute classes (_12b)
- C) For the combination of 1 & 2 above specify the MR's that apply.

New Tables

Table: FLTR2MST

Intr_az_id n(4,0): Code for _12a zone name
Int_class n(3,0): internal code relating to table
FLTR2CLS
Int_Mr n(3,0): internal code relating to table
fltr2mr

Table: FLTR2CLS

INT_CLASS n(3,0): internal code relating to table
FLTR2MST
Int_code: _12b internal class code

Table: FLTR2MR

INT_MR n(3,0): internal code relating to table
FLTR2MST
In_mgmr_id n(3,0): internal code for the selected
management regime

Screen title - Management Regime Zone Filter

New Zone	Delete Zone	Done
----------	-------------	------

Previously
Defined
Zone(s)

OK	Abort
----	-------

Zone Class Names

Candidate

Selected

Management Regime Names

Candidate

Selected

New Zone push button lists _12as zone names for cutting units (C,LC) that are not already in table FLTR2MST.intr_az_id. Store intr_az_id to FLTR2MST.intr_az_id and create int_class and int_MR codes.

Delete Zone removes the current zone from the list of defined zone list, FLTR2MST and delete the corresponding record(s) from both FLTR2CLS and FLTR2MR base on FLTR2MST.INT_CLASS and FLTR2MST.INT_MR.

The zone class Names are the corresponding _12b.name for the current _12a.intr_az_id. The management regimes names come from _43b for those with path true. Once the user starts to select or deselect zone class names or MR's do not let them leave before clicking the Ok or Abort push button. When OK is chosen, store the selected _12b.int_code to FLTR2CLS.int_code with the current INT_CLASS code. Also store in_mrg_id to FLTR2MR along with the int_mr code. If abort is selected, give a warning message and leave without updating anything.

Item 2.6 Program the Stand Data Pathway Screen

Copy the old screen code for _16 to new screen _16p with the title "Stand Data - Pathway". The new table structures are;

Structure for table: path_mst.dbf

Field	Field Name	Type	Width	Dec
1	STAND_ID	Character	10	
2	HBTY_GRP	Character	8	
3	DOM_SP	Character	8	
4	SZ_CLASS	Character	8	
5	DENSITY	Character	8	
6	TIME_INC	Numeric	4	

Structure for table: _16p.dbf

Field	Field Name	Type	Width	Dec
1	STAND_ID	Character	10	
2	INT_STATE	Numeric	4	0
3	HBTY_GP_I	Numeric	4	0
4	DOM_SP_I	Numeric	4	0
5	SZ_CLASS_I	Numeric	4	0
6	DENSITY_I	Numeric	4	0
7	TIME_INC	Numeric	4	0

Both standard and Path use Std_mstr.

Change the push buttons from old screen to new screen and update list of valid code from _12b.

Forest Type -> Dominate Species

Stocking -> Density

Successional Stage -> Size Class

Add new push button

Habitat Type Group -> Hab Type Group

Time Increment -> Time Inc (1 - 10)

Delete the following items

% Vol by Species, Age, Volume, Basal Area, DBH, Height and habitat type.

Include checking in importation routine for new fields. If the combination of int_hbgp, int_dsp, int_szcl, int_dens cannot be found in Def_state table write error statement. Check the complete listing of stand id's in _17 against both path_mst and std_mstr. If not there write error message.

3. Modify Current Screens and Data Tables

Item 3.1 Update tables _12a, _12as, _12b, anex_12b, anex_12a

Update tables based on imported State data.

Item 3.2 Screen _32

Add option for selecting Pathway Only, Standard Only, If screens are related to path or standard gray-out based on the selected option.

Item 3.3 Screen _21

Add field _21.Path,logical. User must specify which outputs are path and which are standard. Filter variables based on those available for standard versus path for selecting attributes for an output. Outputs related to path are available for resource projects only.

Item 3.4 Screen _29

Add field _29.Path,logical. User must specify which costs are path and which are standard. If a cost is standard only allow the user to select attributes that are from std_mstr, std_attr, and system values. Select only attributes that are available for pathway from path_mst.

Item 3.5 Screen _39

Add field _43b.Path,logical. To old screen _39 add new title "Management Regime - Standard" and Management Regime with option for Standard or Pathway to the main menu item.

Item 3.6 Screen _42

Add a filter to only process management regimes with _43b.path = false.

Item 3.7 Screen _40

Add a filter to only process management regimes with _43b.path = false.

Item 3.8 Screen _49 / Updated Table _49a

If standard, the list of valid acreage controls are the same. If path only or both add the following to list of valid Acreage Controls: State, Dominate Species, Size Class or Density. A_L_NAME codes are ST for state, DSP for Dominate Species, SZC for Size Class, and DNC for Density. If the user selects one of these options a selection/de-selection list in a window (i.e. like Successional Stage) should appear. Store the selected class internal codes to _49a.gen_int_Id with a _49a.rec_type = 7.

Item 3.9 Screen 16

Add "- Standard" to title of screen old _16, Stand Data - Standard. Add new checking for when "Both" option is selected in _32, that looks at both std_mstr and path_mst for complete listing of stands present (to the importing process of stand data).

Item 3.10 Screen 18

Add field to _18 type c(1) with valid codes 'S', 'P' or 'B'. Check each cutting unit from _18 to make sure that all the stands are either "Standard", "Pathway" or "both" (if both check for complete data sets for path and standard). Store type ('S' for "Standard", 'P' for "Pathway", 'B' for "Both") to _18. Report the treatment Unit Id's and Stand Id's which only have a partial list in one group or the other. Add push button for Pathway Management Regime Generator to screen 18.

Item 3.11 Screen _45

Add to the bottom a filter to display costs based on pathway and standard for table 45 (cost).

Item 3.12 Screen 46

Filter the list of cutting unit id's to determine the vegetative projection method ("standard" or "pathway") based on std_mstr ("standard") and path_mst ("pathway"). Based on treatment unit projection type ("standard", "pathway" or "both") only allow management regimes of the same type. Add to display for current cutting unit Standard (), Path () or Both ().

Check for super cutting units: all cutting unit members must have a complete set of stand data for either path or standard.

Item 3.13 Screen _47

Add to the bottom a filter to display outputs based on pathway or standard for Table 47 (no activity outputs).

Item 3.14 Screen _48

Add to the bottom a filter to display outputs based on pathway and standard for table 48 (activity output).

Item 3.15 Expand the Capability of the Equation Builder and Associated Routines

Expand the capability of the equation builder to handle:

- a. equations that compute price for outputs stored in _21 for use in the pathway approach, and
- b. equations that compute quantity for a specified output.
(These outputs, however, are different from those listed in _21)

There would have to be a flag to identify price equations from output equations.

The variables that could be used in both of these equations as the independent variables are:

- a. Any output listed in _21 for use in the pathway approach, including the 4 system variables SYS_DBH, etc.
- b. Any cost item in _29 for use in the pathway approach
- c. Any stand attribute that is continuous (type = continuous in _12a)

The screen(s) for building equations will require some modifications (specify equation as price or output and provide a name for the output to be predicted or priced). Also, the data structures for storing the equations may have to be modified to accommodate these two type of equations. The code for applying the equations will need to be modified to accommodate the list of possible independent variables listed above, but once the values for these variables are determined, plugging them into the equation should be the same process.

Item 3.16 Screen exec_sps

Add a filter to only process management regimes with _43b.path = false.

4. Construct New Batch Processes

Item 4.1 Pathway Management Regime Generator

A. Routine to Create No Action State by time unit.

```
Create No Action State by time unit
select pro_state
set filter to activity='SUCCESSION'
create table no_st_tu (hbty_gp_i, st_tu1, ti1, st_tu2, ti2,
st_tu3,ti3,st_tu4,ti4,st_tu5,ti5, st_tu6,ti6,st_tu7,ti7,
st_tu8,ti8,st_tu9,ti9,st_tu10,ti10) where hbty_gp_i is internal habitat
type group, st_tu1 is state for time unit 1, ti1 is time increment for
time unit 1, ...
*
store the internal code for 'SUCCESSION' to m.act
* get a complete set all combination of state, habitat type
* group and time increment into state
select distinct state, time_inc, hbty_gp_i from _16p into array a_state
store _tally to m.cnt
* loop through a_state (all combination of state, habitat
* type group and time increment into state)
for icomb = 1 to m.cnt
  store a_state(icomb,3) to m.hbty_gp_i
  store a_state(icomb,2) to m.time_inc
  store a_state(icomb,1) to m.state
  select no_st_tu
  append blank
  replace no_st_tu.hbty_gp_i with m.hbty_gp_i
  for itinc = 1 to 10
    replace eval("no_st_tu.st_tu"+ alltrim(str(itinc))) with m.state
    replace eval("no_st_tu.ti_"+ alltrim(str(itinc))) with m.time_inc
    select pro_state
    seek instate+hbty_gp_i+time_inc+m.act
    store pro_state.instate to m.state
    store pro_state.Rtime_inc to m.time_inc
  next itinc
next state+time_inc
```

B. Create a list of distinct zone ids from FLTR2MST.INTR_AZ_ID. Store to array dis_zone.

C. Create an array dis_mr of distinct int_mr from fltr2mr. For a management regime in dis_mr determine the valid list of FLTR2MST.intr_az_id and FLTR2CLS.int_code (zone class where this management regime can be applied). Check to make sure that there is at least one valid combination of fltr2mst.intr_az_id and fltr2cls.int_code for each distinct fltr2mst.intr_az_id code. If for a given zone there is not a valid zone class name (int_code) to apply to this management regime write warning message saying "Management Regime: XXXX is was not used because it is not specific for the following zones: _12as.name.", do not process management this regime,

and skip to next management regime. Determine the list of valid combinations of state, habitat type groups, and each time increment that falls between min_tu and max_tu based on data from table _43p, _43pa and _43pb for the current management regime (store v_fl2).

- D. For each treatment unit that is pathway, check the cutting unit int_code (table anex_18) for the distinct zone id (dis_zone) against the valid list created in step 3. IF there is a match for each zone (i.e. at least one class per zone matches) then continue otherwise skip to the next cutting unit.
- E. Determine the state and time increment for the current treatment unit at the mid point time unit of each time period. Loop through all stands in the treatment unit. Use _16p.intstate, _16p.time_inc, and _16p.hbty_gp_i to locate record in no_st_tu. Store to temp table for the mid point time unit of each time period, st_tu+(midpoint time unit), ti+(midpoint time unit) and the number of acres in stand. If the combination of state and time increment are already in temp table, add current stand acres to the acre value. For a given time period, delete all but the row with the maximum acres.
- F. For all time periods determine list of valid time period by comparing the valid list of state, habitat type group, and time increment (min_tu and max_tu) with state, time increment and time are stored to temp_array for this cutting unit. If the temp.state, _16p.hbty_gp_i, and temp.time_inc are valid in v_fltr2, store the temp_array.time period to a_valid_tp.
- G. Determine from cu_mngt the default logging method, loading type, and node. Select all the records from cu_mngt where the cu_mngt.cut_un_id = cur cutting unit and ex_mr_rg and fu_mr_rg = DEFAULT or default to temp table. Replace ex_mr_rg with the current management regime and the fu_mr_rg with "NO_ACT". Update the time period in _46 to be only the time period in a_valid_tp. Use the existing import code to import into _46 and _46b.

Item 4.2 Batch process to build tables _50a, _50b, _50c, and _54

A. Intermediate tables that hopefully will speed processing

1. Table (or array) MR_STATE

<u>Field</u>	<u>Type</u>	<u>Description</u>
INT_MR	N(3,0)	Int Mgt Regime code from _43b
INT_ACT	N(3,0)	Int Activity code from _37
TU_NEXT_ACT	N(2,0)	Number of TUs until the next activity in the Mgt Regime (calculate from _39) For the last activity in the Mgt Regime enter the number 10 to represent through the last time unit in the planning horizon

2. Table (or array) RELEVANT

<u>Field</u>	<u>Type</u>	<u>Description</u>
INT_OUT	N(3,0)	Int output ID from _21
INT_ACT	N(3,0)	Int Activity code from _37, that is found in _48 or as an associated activity in _48a.

This table identifies the activities within each Mgt Regime that affect the quantity of each output, and the amount of TUs since the previous activity in the Mgt Regime that affects the output.

To build this Table: For each output (_21) for each Mgt Regime applying to Path, loop through activity with the Mgt Regime. If the activity code is present in _48 for the current activity, then the activity affects that output. Write a record to RELEVANT where TU_NUM is the number of TUs since the previous activity in the Mgt Regime that affects the current output. TU_NUM = 0 for the first activity in the Mgt Regime. If the activity is present in _48a as an associated activity that has the same output response as its pointer activity, put that associated activity in INT_ACT.

3. Table (or array) STD_STATE

<u>Field</u>	<u>Type</u>	<u>Description</u>
TU	n(2,0)	Time unit
STD	10C	Stand ID for stand that is a member of the current treatment unit
STATE	N(4,0)	Internal code for State
TIME_INC		Time Increment within the State

This table stores the State that is present for each STD present in a treatment unit for each time unit for a specific resource project (a particular impl period in a _46 record.)

4. Table (or array) NO_STATE

<u>Field</u>	<u>Type</u>	<u>Description</u>
STD_ID		Stand ID
TP		Time Period
END_STATE		STATE present and end of the time period for NO ACTION
TIME_INC		Time increment for STATE at the end of the time period for NO ACTION

This table stores the STATE and time increment that are present at the end of each time period for NO ACTION. Used in calculating the starting STATE when implementation period > 1

5. Table (or array) STATE_MAX

<u>Field</u>	<u>Type</u>	<u>Description</u>
INTSTATE		STATE present and end of the time period for NO ACTION
INT_HBGP		Int code for habitat group
MAX_TU		Length of STATE in TUs

This table stores the length of the STATE when succession kicks-in and causes STATE to change. To build this table select TIME_INC from table PRO_STATE where INTSTATE = STATE and INT_HBGP = the habitat group for the stand (_16p).

6. Table (or array) STD_OUTPUT

<u>Field</u>	<u>Type</u>	<u>Description</u>
STD		Stand ID
TU_ACT1		Time units from the first activity
XXXXXXX		Quantity of output XXXXXX, where XXXXXX represents a particular output (maybe the field name for XXXXXX should be the output ID?) There will be one of these fields for each active output in the problem, including the system outputs SYS_DBH, SYS_VOL, SYS_DEN, and SYS_HT.

This table is built for each resource project. The output quantities are running totals across the stands in the treatment unit for each time unit following the first activity. Routine COMPUTE_OUTPUTS is called each time 1) another activity is reached in the Mgt Regime, 2) each time State changes due to Succession. Each time it is called, additional cells in STD_OUTPUT are filled-in. That means that this table must stay around through multiple calls to the routine, and be re-initialized when a new resource project is processed.

7. Table (or array) OUTPUT_TU_START

<u>Field</u>	<u>Type</u>	<u>Description</u>
OUT_ID		Output ID
TU_48		The TU to use for the first output amount to get from _48

This table is built for each resource projects. it is used in each call to routine COMPUTE_OUTPUTS so the data must be maintained until a new Resource Project is to be processed, at which time the table is re-initialized.

Time Periods and time units in problem:

TP1	TP2	TP3	TP4
-- -- --	-- -- --	-- -- --	-- -- --
1 2 3	4 5 6	7 8 9	10

↑
impl period=2 (first activity at begin of TU4)

STD_OUTPUT								output #1
								output #2
	1	2	3	4	5	6	7	TU_ACT1

8. Table (or array) std_out_no

Field	Type	Description
std_id		stand id
cut_time_unit		time unit, where the first time unit in period 1 is time unit 1
iout_numb		number or code associated with an output (there will be one of these for each output in _21 associated with the path approach and the 4 system outputs, SYS_DBH, etc.

9. Modify _50c

Add a field _50c.path, logical for identifying which records are associated with the path approach _50c.path = T, and which records are associated with the standard approach _50c.path = F.

B. Procedure up_state

```
parameter m.hbty_gp_i, m.time_inc, m.intstate, m.int_act_id
m.time_inc = m.time_inc + 1
* update state
select pro_state
seek m.hbty_gp_i + m.time_inc + m.intstate + m.int_act_id
scatter memvar
store m.restate to m.cur_state, m.intstate
* update component of state
select def_state
seek m.intstate
scatter memvar (these variable are global to all procedure and main
used to as attributes for outputs
```

C. Initialization Routines

1. Routine Create table NO_STATE (part of bld_path initialization)

```
* create file no_st_tu by run Create No Action State
*
select no_st_tu
* (hbty_gp_i, st_tu1,ti1,st_tu2,ti2,
* st_tu3,ti3,st_tu4,ti4,st_tu5,ti5, st_tu6,ti6,st_tu7,ti7,
* st_tu8,ti8,st_tu9,ti9,st_tu10,ti10)
```


IF no_st_tu not built, process Create No Action State by time unit (see notes under pathway management regime generator for creating no action state).

```
* Determine the ending tu for time periods 1,2,3,4 or number of time
  periods - 1 and store to array end_tu (only dimension for the number
  of time periods - 1)
select distinct stand_id,intstate,hbty_gp_i,time_inc from _16p into
  array a_list
store _tally to m.cnt
for i = 1 to m.cnt
  store a_list(i,1) to m.std_id
  store a_list(i,2) to m.state
  store a_list(i,4) to m.time_inc
  store a_list(i,3) to m.hbty_gp_i
  select no_st_tu
  locate for no_st_tu.state = m.state and no_st_tu.time_inc = m.time_inc
    and no_st_tu.hbty_gp_i = m.hbty_gp_i
  if not found()
    write error message to err file
  else
    scatter memvar
    for itp = 1 to elen(end_tu,1)
      store ("m.st_tu"+alltrim(str(end_tu(itp)))) to m.tmp_st
      store ("m.ti_" +alltrim(str(end_ty(itp)))) to m.tmp_ti
      insert into no_state values(m.std_id,itp, eval(m.tmp_st),
        eval(m.tmp_ti))
    next itp
  endif
next i
```

2. Calculating information about the mid-points of the time periods.

a. Calculate the mid-point time unit in period p (MPTU_p)

If the number of TUs in period p is odd:

$$\text{MPTU}_p = \text{integer} (\text{TUNUM}_p/2) + 1$$

If the number of TUs in the period is even:

$$\text{MPTU}_p = \text{integer} (\text{TUNUM}_p/2)$$

where:

TUNUM_p = the number of time units in period p

b. Number of time units from the first TU in period 1 to the mid-point TU in period P (N_p)

$$N_p = \text{MPTU}_p + \sum_{p=1}^{P-1} \text{TUNUM}_p$$

- a. Do this for each equation that predicts price, for each time period

$$\text{PRICE\_FACTOR}_{op} = (1+r_{ot})^h * \left[\prod_{t=1}^{N_p-1} (1+r_{ot})^{LTU} \right]$$

where:

PRICE_FACTOR_{op} = factor for product o to update the price predicted by equation to the mid-point of time period p

N_p = the number of TUs to the mid-point TU for period p

h = number of years from the beginning of the mid-point TU to the actual mid-point year for the time period (see notes in 2 above)

t = counter for TU

LTU = number of years in a TU (table _31)

r_{ot} = rate of increase for output o in time unit t (from tables _33a & _33b)

5. Routine to calculate the costs at the mid-points of the time periods

- a. Create a temporary cost table (_45UP) that is identical to _45, except adds a new field for time period for storing the updated costs.

- b. Loop for each cost item in _29

(1) Loop for each time period

$$(a) \text{ FACTOR} = (1+r_{ct})^h * \left[\prod_{t=1}^{N_p-1} (1+r_{ct})^{LTU} \right]$$

where:

N_p = the number of TUs to the mid-point TU for period p

t = counter for TU

h = number of years from the beginning of the mid-point TU to the actual mid-point year for the time period (see notes in 2 above)

LTU = number of years in a TU (table _31)

r_{ct} = rate of increase for cost c in time unit t (from tables _33a & _33b)

(b) Loop for each record in _45 where internal cost = the current cost

Write record to _45UP

Copy all fields in _45 to _45UP except cost

cost = _45.cost * FACTOR

time period = current time period

D. Loop for each implementation option for each record in _46. These notes assume that all the records pertaining to a specific treatment unit are processed one after another in the loop, and secondly, that the first option processed for a treatment unit is the No Action alternative (Exist Mgt Regime = NO ACTION). Note also, the process requires a no action for super cutting units, but do not write records for this to _54, _50b,c.

Write records to _54

a. Write a type 3 record to _54 for the current Resource Project (impl. period and rest of information in the current _46 record). (Keep the VAR_ID available for writing to _50a-c tables.)

b. Write records to _54 for variable flow loading variables.

If Loading type = "V" for the current _46 record, then:

Write a type 4 record to _54 for each corresponding

_46b.record_type = 1 (matching int_combo_id between _46 & _46b),
and for each traffic type defined in _6.

Endif

* zero tot_cut_ac before each super cutting or before treatment

* unit store 0 to tot_cut_ac

Loop for each treatment unit that is a member of the treatment unit in the current _46 record. (When the _46 treatment unit is a super treatment unit, there will be more than one treatment unit to process in this loop.

When it is not a super treatment unit, i.e. regular treatment unit, there will be only 1 pass through this loop.)

If the previous treatment unit processed in _46 <> to current _46 treatment unit, then:

Initialize std_out_no to zero

1. If: NO ACTION then do this step, otherwise go to step 2.

* check past activity for current cutting unit

if past activity (_18.pas_act_id) <> 0

determine the list of outputs that are affected (l_aff_out) by the past activity.

endif

```

for all stands in current cutting unit
  scatter memvar
  store 0 to m.cur_time_Unit, m.cur_time
  * store the time unit for past activity to l_time
  store round(_18.yr_sin_act/_32.time_len,0) to l_time
  store m.intstate to m.cur_state
  * calculate total treatment unit acres
  tot_cut_ac = tot_cut_ac + _17.stand_ac
  * Loop through all time periods in model
  for itp = 1 to 5 time period (or the number for the current model)
    * For a current time period loop through all the
    * time units in the time period

    for itu = 1 to _31.no_tus_itp
      * increment time unit count for accessing _47
      m.cur_time_unit = m.cur_time_unit + 1
      * increment time unit count for accessing _48
      l_time = l_time + 1
      * Loop through all outputs from _21 that have
      * path true
      for iout_numb = 1 to count of _21.path true
        * use the same structure from
        * BLD_50ac for look up outputs
        * Compare the current out id with listed
        * from l_aff_out which table _48 or _47
        if _21.int_out_id in array l_aff_out
          std_out_no(std_id,m.cut_time_unit,iout_numb) = (_48.output for
            time unit = l_time) * _17.stand_ac +
            out(std_id,m.cut_time_unit,iout_numb)
        else
          std_out_no(std_id,m.cut_time_unit, iout_numb) = (_47.output
            for time unit = cur_time_unit) * _17.stand_ac +
            std_out_no3(std_id,m.cut_time_unit, iout_numb)
        endif
      next output
      if mid_tp(itp) = m.cur_time_unit
        store m.intstate, itp, tot_ac to temp_state table. If table
        doesn't have the current state and time period match add it,
        otherwise add current stand acres to the current row value for
        tot_ac.
        *
        for system output name (SYS_DBH, SYS_VOL, SYS_DBH,SYS_HT)
        determine the iout_numb and for each do the following
        store the amount of std_out_no(std_id,m.cut_time_unit,
          iout_numb) to array name sys_tot(type,time period) where type is
          an index for SYS_DBH, SYS_VOL, SYS_DEN, SYS_HT and time period =
          itp.
        endif
        =up_state(m.hbty_gp_i, m.time_inc, m.intstate, m.int_act_id = the
          code for SUCCESSION)
      next time unit in the time period
    next time period
  next stand in current cutting unit

```



```

call Greg's output routine to calculate outputs & write to _50b
  (routine TRT_OUTPUTS)
for itp = 1 to 5
  Find the record that has the max tot_ac for the given time period itp
  from temp_state. Store temp_state.intstate to m.intstate.
  insert into _50c (int_prj_id, itp, sys_tot(1, itp) / tot_cut_ac,
    sys_tot(2, itp) / tot_cut_ac, sys_tot(3, itp) / tot_cut_ac,
    sys_tot(4, itp) / tot_cut_ac, 0, m.intstate, .T.)
next time period
store 0 to out

```

Go to the top of the loop for processing the next record in _46.

2. Copy _50b and _50c records for NO ACTION for the current treatment unit to _50b and 50c for the current resource project for period of occurrence=1 to 1 less than the impl period (used only when implementation period > 1). Use the VAR_ID written to _54 for the current resource project, all other information is copied field for field.

3. Initialize tables &/or arrays

- a. Table (or array) OUTPUT_TU_START. This can be done once for each resource project, and stored. This stored copy can then be used as the initial values for each stand. (Actually, this would be the same for all resource projects having the same impl. period on a given treatment unit.)

Loop through each output

IF: A past activity is listed for the treatment unit in 18 AND the current output is listed in RELEVANT for that listed activity, Then:

TU_48 for each combination of STD_ID and OUT_ID = 1 + the number of TUs since that past activity (listed in 18 as years, must convert to TUs) + the number of TUs in the time periods prior to the period of impl. period.

ELSE:

TU_48 for each combination of STD_ID and OUT_ID = 1

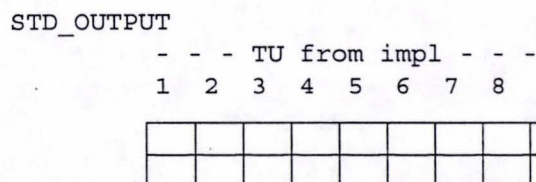
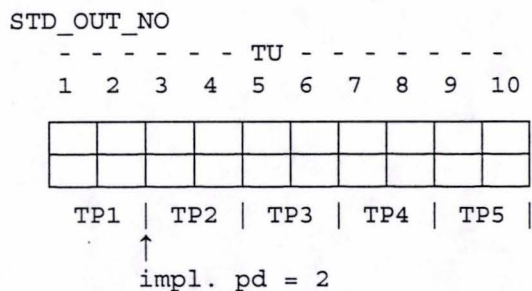
Endif

Endloop

- b. Table (or array) STD_OUTPUT

STD_OUT_NO and STD_OUTPUT both store quantities of output for each output type, by TU, for each stand in the treatment unit. They differ only in that STD_OUT_NO counts TUs beginning with the first TU in time period 1. STD_OUTPUT counts TUs beginning with the first TU in the time period of implementation for a Mgt Regime.

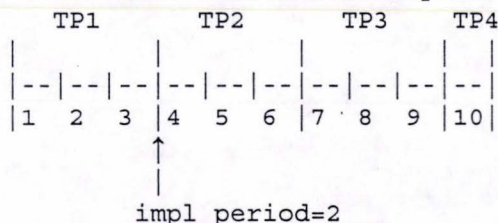
Copy the portion of STD_OUT_NO storing outputs from the first time unit in the period of implementation to the end of the planning horizon to STD_OUTPUT.



So, copy info from TU=3 in STD_OUT_NO to TU=1 in STD_OUTPUT, and so on.

4. Loop for each stand present in the treatment unit referenced in the current _46 record. (This loop processes State for each Resource Project.)
 - a. This step determines and stores State to STD_STATE for the current stand for each time unit, starting with the first time unit in the period of implementation and continuing through the total number of time units used as defined in _31.

Time Period and time units in problem:



Time Units following implementation:



Assumption: For purposes of calculating State, it is assumed that the first activity in the management regime is implemented at the beginning of the time period of implementation.

Initialize Variables:

PREV_ACT = the activity previous to the current activity, get from past activity field in _18. If there are no previous activities listed for the treatment unit, then PREV_ACT = 0

TU_NEXT_ACT = from MR_STATE get TU_NEXT_ACT for INT_ACT = the first activity in the Mgt Regime.

Current State (CUR_STATE):

- a) When the impl. period is 1: CUR_STATE = the beginning State listed in _16p
- b) When the impl. period is 2+: CUR_STATE = the State listed for the previous time period in table NO_STATE

Time increment into the Current State (TIME_INTO_STATE)

- a) When the impl. period is 1: TIME_INTO_STATE = the Time Incr listed in _16p
- b) When the impl. period is 2+: TIME_INTO_STATE = the time increment listed for the previous time period in table NO_STATE

CUR_ACT = The first activity in the Mgt Regime

TU_ACT1 = 1 (TU following first activity)

TU_START = 1 (TU following activity for _48, used to update OUTPUT_TU_START if the activity is in _48)

- (1) Select the next State (NEXT_STATE) and associated time increment (TIME_INC) that results from the current activity, from table PRO_STATE where

INTSTATE = CUR_STATE

TIME_INC = TIME_INTO_STATE

INT_ACT_ID = CUR_ACT

- (a) If a match is found and NEXT_STATE <> CUR_STATE:

CUR_STATE = NEXT_STATE

If: TIME_INC ≥ 1, then:

TIME_INTO_STATE = TIME_INC - 1

Else:

TIME_INTO_STATE = 0

Endif

<--State-->
Comments: |---|---|---|
 1 2 3 (time increments in State)

↑
 At the beginning of the State, TIME_INTO_STATE should equal 0 because no TUs have elapsed. But time increment equals 1, meaning we are in the first TU of the State.

- (b) If a match is not found:

CUR_STATE and TIME_INC stay unchanged

(2) Determine the number of TUs until the next activity in the Mgt Regime, or until Succession kicks-in, which ever is less, and update accordingly.

(a) Calculate TUs until SUCCESSION causes State to change (TU_SUCCESSION).

TU_SUCCESSION = The total number of time units for current State (table STATE_MAX) minus TIME_INTO_STATE

(b) Compare and Update

If: TU_NEXT_ACT $\leq$ TU_SUCCESSION

Write the current State to STD_STATE for the number of TU = TU_NEXT_ACT (starting from TU_ACT1, which is where the pointer is in STD_STATE)

Call Routine COMPUTE_OUTPUTS using following parameters:

(PREV_ACT = PREV_ACT

CUR_ACT = CUR_ACT

TU_ACT1 = TU_ACT1

CUR_STATE = CUR_STATE

TIME_INTO_STATE = TIME_INTO_STATE

TU_START = 1

TU_END = TU_NEXT_ACT)

TIME_INTO_STATE = TIME_INTO_STATE + TU_NEXT_ACT

TU_ACT1 = TU_ACT1 + TU_NEXT_ACT

PREV_ACT = CUR_ACT

TU_START = 1

CUR_ACT = the next activity in the Mgt Regime

Get TU_NEXT_ACT from table MR_STATE

(where INT_ACT = CUR_ACT from previous line)

Continue to Step (3) below

If: TU_NEXT_ACT > TU_SUCCESSION

Write the current State to STD_STATE for the number of TU = TU_SUCCESSION (starting from TU_ACT1, which is where the pointer is in STD_STATE)

Call Routine COMPUTE_OUTPUTS using the following parameters:

(PREV_ACT = PREV_ACT

CUR_ACT = CUR_ACT

TU_ACT1 = TU_ACT1

CUR_STATE = CUR_STATE

TIME_INTO_STATE = TIME_INTO_STATE

TU_START = TU_START

TU_END = TU_NEXT_ACT)


```

TIME INTO STATE = TIME INTO STATE + TU_SUCCESSION
TU_ACT1 = TU_ACT1 + TU_SUCCESSION
PREV_ACT = CUR_ACT
TU_START = TU_START + TU_SUCCESSION
TU_NEXT_ACT = TU_NEXT_ACT - TU_SUCCESSION
CUR_ACT = SUCCESSION
Continue to Step (3) below

```

- (3) If STD_STATE has not been completed through the end of the planning horizon for the current stand, go back to step (1) and process the new CUR_ACT.
ELSE continue to step 5 below.
5. Determine the STATE to be reported for each period for the resource project. It is the STATE most prevalent in the treatment unit at the mid-point TU for each time period.
 - a. Loop for each time period, beginning with the impl. period
 - (1) Loop for each std in the treatment unit

Determine what State is present at the mid-point of period p.
Add the number of acres in that stand to a running total of acres for that State.
 - (2) Write record to _50c for current time period

VAR_ID = matches VAR_ID written to _54 for this resource project
Period of Occur = current time period in loop
DBH = The sum of SYS_DBH across stands in STD_OUTPUTS for the mid-point TU for the current time period in the loop, divided by the total acres in the treatment unit
VOLUME = The sum of SYS_VOL across stands in STD_OUTPUTS for the mid-point TU for the current time period in the loop, divided by the total acres in the treatment unit
DENSITY = The sum of SYS_DEN across stands in STD_OUTPUTS for the mid-point TU for the current time period in the loop, divided by the total acres in the treatment unit
HEIGHT = The sum of SYS_HT across stands in STD_OUTPUTS for the mid-point TU for the current time period in the loop, divided by the total acres in the treatment unit
Int Seral Stage = The STATE with the largest number of acres for the current time period
Int Mgt Regime = Existing Std Mgt Regime from current _46
Path = .T.
6. Call Routine TRT_OUTPUTS to calculate outputs by period for the treatment unit and write to _50b.
7. Compute Costs and Write to _50a
 - a. Loop for each activity in the current Mgt Regime that occurs within or before the last TU in the last period

(1) Loop for each cost item in the current activity (_37a)

(a) If: The units of measure for the current cost item is per acre then:

```
TRT_COST=0
Loop for each stand in the treatment unit
  TRT_COST = TRT_COST + (acres in current stand)*(cost amt
    from _45UP for the current stand for the
    period of the current cost (same as current
    activity))
    (The possible attributes for the cost table
    are the stand attributes in _16P, State (get
    this from either _50c or STD_STATE for mid-
    point TU for the period of occurrence of this
    cost item, Dominate Species, Size Class, or
    Density get from DEF_STATE table where
    INTSTATE = current STATE.)
Endloop
Write record to 50c
  VAR_ID = same as written to _54 for current resource
    proj.
  Period of occur = period for the current cost (same as
    the current activity)
  Int Cost ID = for the current cost item
  Cost/unit = TRT_COST / total treatment unit acres
```

(b) Else If: The units of measure for the current cost item is per some unit of output in _21, then:

```
TOT_TRT_OUTPUT
TRT_COST=0
Loop for each stand in the treatment unit
  TOT_STD_OUTPUT = Using table STD_OUTPUTS sum the quantity
    of all outputs matching the units of
    measure for the cost, across all time
    units within the time period of the cost
  TOT_TRT_OUTPUT = TOT_TRT_OUTPUT + TOT+STD+OUTPUT
  TRT_COST = TRT_COST + (acres in current stand)*(cost amt
    from _45UP for the current stand for the
    period of the current cost (same as current
    activity))
    (The possible attributes for the cost table
    are the stand attributes in _16P, State (get
    this from either _50c or STD_STATE for mid-
    point TU for the period of occurrence of this
    cost item, Dominate Species, Size Class, or
    Density get from DEF_STATE table where
    INTSTATE = current STATE.)
Endloop
Write record to 50c
  VAR_ID = same as written to _54 for current resource
    proj.
```


Period of occur = period for the current cost (same as
the current activity)
Int Cost ID = for the current cost item
Cost/unit = TRT_COST / TOT_TRT_OUTPUT

8. Compute outputs from any equations that predict output quantity and write to _50b.

a. Loop for each equation designed to calculate an output quantity

(1) Loop for each time period

(a) Apply the equation computer portion of the equation builder to calculate the quantity for the current output equation for the current time period. The amounts for the possible variable to plug into the equation are determined as follows:

Outputs in _21: Get the output quantity per acre for the treatment unit, for the current time period from _50b, unless the output is a system output, in which case they come from the appropriate field in _50c.

Costs in _29: Get the cost per unit for the treatment unit, for the current time period from _50a

Continuous Stand attribute: Calculate a weighted average for the attribute across the stands in the treatment unit, where the weights are based on the stand acres

(b) Write record to _50b

VAR_ID = same as written to _54 for resource project
Period of occur = the current time period in loop
int output id = id of the output the equation predicts
quantity/acre = quantity calculated by equation
Price = (leave blank)

9. Compute price from any equations that predict price and update the appropriate _50b records with the computed price

a. Loop for each equation designed to calculate price

(1) Loop for each time period

(a) Apply the equation computer portion of the equation builder to calculate the price for the output specified for the equation for the current time period. The amounts for the possible variables to plug into the equation are determined as follows:

Outputs in _21: Get the output quantity per acre for the treatment unit, for the current time period from

_50b, unless the output is a system output, in which case they come from the appropriate field in _50c. (Note, the outputs can include those computed by equations)

Costs in _29: Get the cost per unit for the treatment unit, for the current time period from _50a

Continuous Stand attribute: Calculate a weighted average for the attribute across the stands in the treatment unit, where the weights are based on the stand acres

(b) Update Price in _50b record where

VAR_ID = same as written to _54 for resource project
Period of occur = the current time period in loop
int output id = id of the output for which the equation predicts

The Updated Price value = $EQN\_PRICE * PRICE\_FACTOR$

where:

EQN_PRICE = price predicted by the equation

$PRICE\_FACTOR_{op}$ = price factor for output o in time period p

E. Routine: COMPUTE_OUTPUTS

1. Parameters passed to routine

a. PREV_ACT = the activity that occurred prior to the current activity

b. CUR_ACT = the current activity, (the activity for which the post activity outputs are calculated). If the activity is an associated activity in _48a, then must use the pointer activity in _48a to look up output amounts in _48.

c. TU_ACT1 = time units from the first activity

d. CUR_STATE = current state

e. TIME_INTO_STATE = when the current activity is implemented in the current state

f. TU_START = identifies which output quantity to start with in _48 -- it is the number of TUs following an activity

g. TU_END = identifies which output quantity to end with in _48 -- it is the number of TUs following an activity

2. Loop for each output including the system outputs SYS_DBH, SYS_VOL, SYS_DEN, and SYS_HT.

Comments: This loop will write output quantities to STD_OUTPUT, beginning with the TU_ACT1 associated with the CUR_ACT (the value of TU_ACT1 passed to this routine) to the last TU_ACT1 that falls within the planning horizon.

- a. Check table RELEVANT to determine if there are records in _48 for the current output type and CUR_ACT.

- (1) If not found in RELEVANT, then do the following

- (a) Update OUTPUT_TU_START for the current output

$TU\_48 = TU\_48 + TU\_END$

Go to the top of the loop and process the next output.

(NOTE: OUTPUT_TU_START is being updated for each stand, but really it only needs to be updated by treatment unit. The problem is the looping structure in the notes does not seem to accommodate calculating this only one for the treatment unit.)

- (2) If found in RELEVANT, then do the following.

- (a) Update OUTPUT_TU_START for the current output and stand

$TU\_48 = TU\_START$

- (b) Select output quantities from _48 (or some pre-processed version of _48) multiply them by the acres of the current stand, and add them into STD_OUTPUT.

-- The output quantities used out of _48 go from OUTPUT_TU_START.TU_48 to the number needed to fill-in values through the last TU in the last time period.

-- The output from _48 goes into STD_OUTPUT at the location identified by the value in TU_ACT1, and subsequent TUs follow sequentially.

Assume the current output is Output 1, and that TU_START = 1, and TU_ACT1 = 4. Then the values to get out of _48 are identified by X below:

_48 record:

---- TU following Activity ---

1 2 3 4 5 6 7 8 9 10

X	X	X	X	X	X	X	X	X	X
---	---	---	---	---	---	---	---	---	---

Multiply these by stand acres and store them in STD_OUTPUT

----- TU_ACT1 -----									
1	2	3	4	5	6	7	8	9	10
Y	Y	Y	X	X	X	X	X	X	X

(The values represented by Y pertain to TUs prior to the period of implementation are not overwritten. However, values previously stored for TU4 on were overwritten in this step.)

-- The attributes used for the current output in _48 are specified in _21. The following is a listing of where to get values for the current activity (always needed in _48) and the other attributes which may be referenced in _21:

- * Activity for _48 = CUR_ACT passed to the routine
- * State = CUR_STATE passed to the routine
- * TIME_INC = TIME_INTO_STATE passed to the routine
- * Previous activity = PREV_ACT passed to the routine
- * Dominate species class, size class, or density class = get from table DEF_STATE where INTSTATE = CUR_STATE
- * User-specified stand attributes = get from ANEX_16 table

F. Routine TRT_OUTPUTS

1. Loop for each output in _21 used in the path approach, except the 4 system outputs SYS_DBH, etc.
 - a. Loop for time period, starting with the period of implementation for the current resource project
 - (1) Calculate TRT_{op} : Sum the current output stored in STD_OUTPUTS across the stands in the current treatment unit and across the time units in the current time period.
 - (2) Write record to _50b

VAR_ID = matches ID written to _54

Period of occurrence = current time period

Internal output ID = current output

output quantity/acre = TRT_{op} / total treatment unit acres

price = UP_PRICE_{op}

Item 4.3 Updates to Matrix Generation Routine

A new data field is added, `_50c.path` (T/F) (see item 4.2, section A.9). To the current code for processing `_49`, for A/L type SST, add skip for `_50c.path = T`. There are 4 new A/L types associated with the path approach to process in `_49`: ST for state, DSP for Dominate Species, SZC for Size Class, and DNC for Density. Use the same rules for State as used for SST, but only process with `_50c.path = T`. If the user has created rows using Dominate Species (DSP), Size Class (SZC), or Density (DNC) use the current `_50c.instate` to look up in table `Def_state` to get the current value, and see if there is a match with the user selected classes (`_49a.int_code`).